Journal of Computer and System Sciences **59**, 412–449 (1999)
Article ID jcss.1999.1636, available online at http://www.idealibrary.com on IDEAL®

Models of Nondeterministic Regular Expressions[1]

Flavio Corradini

Dip. di Mat. Pura ed Appl., Università dell'Aquila, L'Aquila, Italy
E-mail: flavio@univaq.it

Rocco De Nicola[2]

Dip. di Sistemi e Informatica, Università di Firenze, Florence, Italy
E-mail: denicola@dsi.unifi.it

and

Anna Labella

Dip. di Scienze dell'Informazione, Università di Roma "La Sapienza," Rome, Italy
E-mail: labella@dsi.uniroma1.it

Received October 21, 1996; revised March 15, 1999

Nondeterminism is a direct outcome of interactions and is, therefore a central ingredient for modelling concurrent systems. Trees are very useful for modelling nondeterministic behaviour. We aim at a tree-based interpretation of regular expressions and study the effect of removing the idempotence law $X+X=X$ and the distribution law $X\bullet(Y+Z)=X\bullet Y+X\bullet Z$ from Kleene algebras. We show that the free model of the new set of axioms is a class of trees labelled over A. We also equip regular expressions with a two-level behavioural semantics. The basic level is described in terms of a class of labelled transition systems that are detailed enough to describe the number of equal actions a system can perform from a given state. The abstract level is based on a so-called *resource bisimulation preorder* that permits ignoring uninteresting details of transition systems. The three proposed interpretations of regular expressions (*algebraic*, *denotational*, and *behavioural*) are proven to coincide. When dealing with infinite behaviours, we rely on a simple version of the ω-induction and obtain a complete proof system also for the full language of nondeterministic regular expressions.

[1] This work has been partially founded by EEC within the HCM Project EXPRESS, and by CNR within the project "Specifica ad Alto Livello e Verifica di Sistemi Digitali."

[2] Corresponding author.

0022-0000/99 $30.00

1. INTRODUCTION

The theory of regular languages was first studied by Kleene [23] and then axiomatized by Salomaa [30] to obtain so-called Kleene algebras. These are algebraic structures with $+$, $\bullet$, *, 0, and 1 operators satisfying certain properties (the reader is referred to Table 1 for a first sample) that have been fruitfully used also in many areas of computer science.

Building on an alphabet A, regular expressions can be defined via the syntax below:

$$P ::= 0 \mid 1 \mid a \mid P + P \mid P \bullet P \mid P^*, \quad \text{where } a \text{ is in } A.$$

We will refer to the set of terms generated by this BNF as **PL**, for process language.

Recently a new axiomatization of Kleene algebras has been proposed by Kozen [25] (see also [6]), that relies on the original axiomatization of Table 1 proposed in [30] for finite (*-free) terms, and on the laws of Table 2 for infinite expressions. There, $\leqslant$ stands for the partial order obtained by asserting $X \leqslant X + Y$ and the axioms in Table 1 and by requiring preservation by $\bullet$ and $+$.

Regular expressions and Kleene algebras have also been a direct inspiration for many of the constructs and axiomatizations of concurrency models such as CCS, CSP, and ACP (see, e.g., [9, 22, 27]), generally referred to as *process algebras*. If one considers **PL** as defined above, it is possible to interpret its operator symbols in terms of basic agents and operators for agents composition. Thus 0 can be seen as the zero agent introduced in [5], 1 as the successfully terminating one, and a as the agent that executes action a and then successfully terminates. Moreover, $+$ can be seen as the operator for nondeterministic compositions of agents and $\bullet$ as the operator for their sequential composition.

The main differences between the axiomatization of finite regular expressions and those for process algebras are essentially due to the different stresses that process

TABLE 1

Complete Set of Axioms for Finite Regular Expressions

$X + Y = Y + X$	(C1)
$(X + Y) + Z = X + (Y + Z)$	(C2)
$X + 0 = X$	(C3)
$X + X = X$	(C4)
$(X \bullet Y) \bullet Z = X \bullet (Y \bullet Z)$	(S1)
$X \bullet 1 = X$	(S2)
$1 \bullet X = X$	(S3)
$X \bullet 0 = 0$	(S4)
$0 \bullet X = 0$	(S5)
$(X + Y) \bullet Z = (X \bullet Z) + (Y \bullet Z)$	(RD)
$X \bullet (Y + Z) = (X \bullet Y) + (X \bullet Z)$	(LD)

TABLE 2

Axioms for *

$1 + X \bullet X^* \leqslant X^*$			(*1)
$1 + X^* \bullet X \leqslant X^*$			(*1d)
$Z + X \bullet Y \leqslant Y$	implies	$X^* \bullet Z \leqslant Y$	(*2)
$Z + Y \bullet X \leqslant Y$	implies	$Z \bullet X^* \leqslant Y$	(*2d)

algebras put on nondeterminism. Indeed, the possible structure induced by the $+$ operator is ignored by the traditional interpretation (as sets of strings) of regular expressions. For this interpretation, a distributivity law permits lifting the $+$ at the top level. In the framework of process algebras, since nondeterminism is a direct outcome of interactions and, thus, central to any theory of communicating agents, the same distributivity law cannot be kept in.

In this paper, by following the process algebraic interpretation suggested above, we study the effect of removing the two axioms of finite Kleene algebras that, to some extent, lead to ignoring the nondeterminism implicit in the syntax. The two axioms permit ignoring the fact that specific choices are determined by specific actions; thus, $a \bullet (b + c)$ and $(a \bullet b) + (a \bullet c)$ are considered as equivalent despite the fact that in one case the choice is taken *after* a, while in the other it is performed *before* a. Indeed, axiom (LD) permits considering all regular expressions as denotations of multisets of sequences of actions (traces); and axiom (C4) leads to further flattenings by permitting the elimination of duplicated traces.

We will thus look for denotational and operational semantics of **PL** that are in full agreement with the set of axioms of Kleene algebras once the idempotence law for $+$ (C4) and the distribution law of $\bullet$ over $+$ (LD) are removed from Table 1 (see Table 3). The elimination of (C4) and (LD) is essential for obtaining a tree-based semantics of regular expressions.

We will also consider the nondeterministic infinite behaviours induced by the * operator and their impact on the rules of Table 2 proposed by [25]. In that table, we can see that there are two pairs of dual rules. This duality plays a crucial rôle in Kozen's completeness proof, but it will be lost once we move to (tree-based) nondeterministic interpretations.

Before moving on to describing the content of this paper more precisely, we would like to provide motivations for the presence or absence of some laws from the axiomatization we consider.

The necessity of avoiding the distributivity axiom (LD) when describing interactive (deadlock sensitive) systems is well known. The two expressions

TABLE 3

Unwanted Axioms

$X + X = X$	(C4)
$X \bullet (Y + Z) = (X \bullet Y) + (X \bullet Z)$	(LD)

TABLE 4

Basic Preorder

$X \leqslant X + Y$	(PR)
$X \leqslant Y$ and $X' \leqslant Y'$ imply $X \bullet X' \leqslant Y \bullet Y'$	(R1)
$X \leqslant Y$ and $X' \leqslant Y'$ imply $X + X' \leqslant Y + Y'$	(R2)

$a \bullet (b + c) = a \bullet b + a \bullet c$ are clearly language equivalent, as they both describe the set of strings $\{ab, ac\}$. However, when viewed as processes they have to be distinguished. While the left-hand side can always perform an action b after action a, the right-hand side has the possibility of refusing to do so. Thus, if we assume that these two processes can interact with their environment, it is essential to consider them as semantically different.

Axiom $X + X = X$, instead, is present in all process algebras axiomatizations. Nevertheless, if one wants to preserve the richness of the syntactic structure and, at the same time, be faithful to tree models, then he may want to leave (C4) out. An independent, more semantic, motivation for eliminating the axiom stating idempotence of $+$ is the interest in formalizing fault-tolerant systems. As a simple example consider process $a + a$ corresponding to the nondeterministic composition of two processes that can perform an a action and then successfully terminate. If a hardware fault leads to shutting down one of the processes (say a printer) of $a + a$ then it would still offer the expected behaviour (a printout would be obtained from the *alternative* printer). The same cannot be said for process a; in fact, a fault of the system where a is located would be noticed (no printout would be obtained). Then, one would say that $a + a$ is more tolerant to faults than a, in the sense that it takes advantage of the different instances of the available resources.[3]

Another axiom, worthy of note, is $X \bullet 0 = 0$ that reduces to 0 all those agents that eventually reach a deadlocked state. This law is not present in the axiomatizations of process algebras (an exception is [5]), but it is commonly used in formal languages and automata theory, where a word is "accepted" by an automaton only if it allows a transition from the initial state of the automaton to a final one. This means that, if a deadlocked state occurs before reaching the final state, the whole computation is ignored. If one wants to take into account the sequence of actions performed before reaching a deadlocked state, then he can take advantage of the possibility of writing, at specification time, $P \bullet (1 + Q)$ instead of $P \bullet Q$. This would permit "accepting" also the sequences of actions performed by P.

The rest of the paper is organized as follows. In Section 2, we will study a denotational semantics for finite (*-free) regular expressions. We will see that the free model of the new set of axioms is equivalent to a class of trees labelled over A. Trees are seen as sets of labelled runs (sets of traces) plus some information about

[3] This kind of fault tolerance is known in the literature as "cold redundancy": different inactive copies of the same process are maintained, and no form of restart is assumed; as soon as an alternative is chosen the others are immediately discarded.

TABLE 5

ω-Induction Rule

Let	$X_0 = 1$ $X_{n+1} = 1 + X \bullet X_n$	
$\forall n \in N$	$X_n \bullet Z \leqslant Y$ implies $X^* \bullet Z \leqslant Y$	(ω-R)

the branching structure of the runs. This permits us to naturally transfer techniques and results developed for formal languages and regular sets to models of nondeterminism.

In Section 3 we will introduce an operational semantics for finite (*-free) regular expressions; it is based on a class of labelled transition systems detailed enough to describe the number of equal actions a system may perform when in a given state. On the top of this operational semantics, we will introduce a preorder relation that we call *resource simulation* (*r-simulation*) and an equivalence relation that we call *resource bisimulation* (*r-bisimulation*). This equivalence will allow us to identify all and only those regular expressions that denote *isomorphic trees*. An important property of the equivalence and the preorder (that will permit us to restrict attention to the latter) is that r-bisimulation can be obtained as the kernel of r-simulation. We study in full detail the preorder $\lhd_r$ induced by r-simulation; we shall prove that the set of axioms obtained from Table 1 by removing (C4) and (LD) and adding the laws of Table 4 is consistent and complete with respect to $\lhd_r$. Of course, in Table 1, any equation $T_1 = T_2$ should be read as $T_1 \leqslant T_2$ and $T_2 \leqslant T_1$.

The complete axiomatization of r-bisimulation equivalence is obtained by removing $X \leqslant X + Y$ from that of the preorder.

In Section 4 we study the relationships between operational and denotational semantics and prove their coincidence.

Section 5 is dedicated to studying the impact of enriching the language with the *-operator and, thus, to considering the induced infinite behaviours. To avoid considering infinite sequences of 1-actions (essentially internal chattering), we restrict attention to terms without iterations of 1's; i.e. we exclude terms with 1 or * summands in a *-context. This essentially amounts to saying that we permit inserting in $[\_]^*$-context only those terms that do not have the empty word property, as defined by Salomaa [30].

As we have already mentioned, not all theorems of [25] are sound for our interpretation. We have that axiom (*1d) does not hold,[4] and that inference rule (*2d) of Table 2 is vacuously true; its premise holds only if $Z = 0$ and $X = 1$.[5] For estab-

[4] Processes $1 + a^* \bullet a$ and a^* are trace equivalent but they are not equivalent for our interpretation because the term on the l.h.s. can execute two initial a-actions while that on the r.h.s can execute only one.

[5] It could be of interest to know that, results in [10] (based on [Kro91]) imply that Kozen's axiomatization is complete for the equational theory of regular sets also when laws (*1d) and (*2d) are removed from the proof system.

lishing our completeness result, we replace (*2) with a more powerful ω-induction rule (see Table 5). The problem of establishing whether this is necessary is still open.

We rely on the correspondence between approximants of terms and those terms that are built by unfolding via the rewriting rule

$$P^* \to 1 + P \bullet P^*.$$

For the denotational semantics, these approximants allow us to build the interpretation of P^* as a colimit and to obtain a complete proof system for the enriched language.

In the final section, we discuss extensions of the language with a binary operator for parallel composition and a complete axiomatization also for this richer language and discuss further work on logical characterization and a weak version of resource bisimulation.

2. FINITE DENOTATIONAL MODELS

In this section we provide a denotational semantics of finite (*-free) **PL** by interpreting it over a category of labelled trees and show that it coincides with the initial model induced by a simple set of equations. For the full understanding of this section a basic knowledge of a few notions of category theory is required. To this purpose the reader is referred to an introductory book; see, e.g., [29] and references therein.

Our category of trees (see [14, 15, 24]) will be named **T**. A single tree will be modelled by listing all of its runs (or paths) and then saying where they agree. Thus, the tree that describes a choice between the two sequences of actions $a \cdot b$ and $a \cdot c$, usually denoted by the term $a \bullet b + a \bullet c$ [27], and represented as in Fig. 1, will be modelled via two runs, x and y, that are labelled by ab and ac, respectively, and have *empty* agreement. In contrast, the tree denoted by $a \bullet (b + c)$, representing the possibility of executing an a and then performing the choice between b and c, and pictured in Fig. 2, will be modelled via the same two runs x and y, labelled again by ab and ac, but with agreement equal to a.

Runs are used to describe computations from one state to another, exactly like strings of actions within automata theory. Additional structure is introduced by agreements.

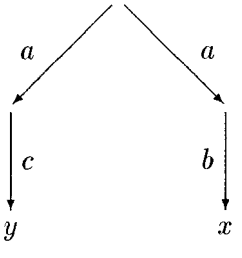

FIG. 1. The tree for $a \bullet b + a \bullet c$.

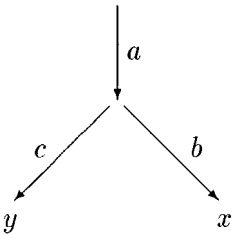

FIG. 2. The tree for $a \bullet (b + c)$

We start by introducing a structure to deal with the labels. Below, $A^\star$ denotes the set of finite strings over the set A.

DEFINITION 2.1. Let $\mathscr{A} = (A^\star, \leqslant, \wedge, \varepsilon)$ be the meet semilattice:

(i) $A^\star$ is the set of words on A,

(ii) $\leqslant$ is the prefix order of words,

(iii) $\wedge$ is the largest common prefix operation on words,

(iv) ε is the empty word.

DEFINITION 2.2. An *$\mathscr{A}$-tree*, that often will be called simply *tree*, $t = (X, \alpha, \beta)$ consists of:

(i) a set X of *runs*;

(ii) a map $\alpha: X \to A^\star$, the *extent map*, giving the computation $\alpha(x)$ performed on a run x;

(iii) a map $\beta: X \times X \to A^\star$, the *agreement map*, saying to what extent two computations *agree*. For the *agreement map* it is required that, for any x, y, z in X,

(a) $\beta(x, x) = \alpha(x)$ (a run agrees with itself along all its length);

(b) $\beta(x, y) \leqslant \alpha(x) \wedge \alpha(y)$ (the agreement between runs is not more than their largest common prefix);

(c) $\beta(x, y) \wedge \beta(y, z) \leqslant \beta(x, z)$ (the agreement between x, y, and z is not more than that between x and z);

(d) $\beta(x, y) = \beta(y, z)$ (it does not matter in what order agreement is specified)

We will write t, t_1, and t_2 for denoting typical trees, with components $t = (X, \alpha, \beta)$, $t_1 = (X_1, \alpha_1, \beta_1)$, and $t_2 = (X_2, \alpha_2, \beta_2)$ A tree morphism from a tree t_1 to a tree t_2 is a map from the set of runs of t_1 to the set of runs of t_2, preserving the extent while allowing the agreement to increase.

DEFINITION 2.3. A *tree morphism* $f: t_1 \to t_2$ is a map $f: X_1 \to X_2$ satisfying

(i) $\alpha_2(f(x)) = \alpha_1(x)$ (f does not change extent);

(ii) $\beta_2(f(x), f(y)) \geqslant \beta_1(x, y)$ (f does not decreases agreement)

We are now set to define our basic category of $\mathscr{A}$-trees and shall denote by **T** the category whose

(i) *objects* are trees ($t = (X, \alpha, \beta)$);

(ii) *arrows* are tree morphisms;

(iii) *identities* ($id_t = id_X$) are defined in terms of identities over set of runs;

(iv) *composition* ($g \circ f$), is given by function composition.

With $\mathbf{T}^{\text{fin}}$ we will denote the subcategory of finite trees.

DEFINITION 2.4. A tree morphism is a *strict* (*or regular*) *monomorphism* if f is injective and $\beta_2(f(x), f(y)) = \beta_1(x, y)$

Note that in $\mathbf{T}^{\text{fin}}$ if $f_1: t_1 \to t_2$ and $f_2: t_2 \to t_1$ are strict monomorphism then t_1 and t_2 are isomorphic. Indeed, we have that a strict monomorphism is an injective tree morphism such that both extent and agreement are preserved. It might be of interest to know that, in categorical terms, strict monomorphisms are *regular monomorphisms*, i.e. *equalizers*.

PROPOSITION 2.5. $\mathbf{T}$ *has an initial object, given by the empty tree* $\mathbf{0} = (\varnothing, \varnothing, \varnothing)$, *and it has coproducts* $\oplus$.

Proof. There is clearly a unique map from $\mathbf{0}$ to any tree t, namely the empty map $\mathbf{0}_t$. For two trees t_1 and t_2, $t_1 \oplus t_2$ is defined as $(X_1 \uplus X_2, \alpha_1 \uplus \alpha_2, \beta_1 \uplus \beta_2)$, (where $\uplus$ denote disjoint set union and $\beta_1 \uplus \beta_2$ denotes the agreement function that behaves as β_1 on pairs from X_1, as β_2 on pairs from X_2, and is ε on mixed pairs) Clearly, the canonical injections

- $i_1: t_1 \to t_1 \oplus t_2$
- $i_2: t_2 \to t_1 \oplus t_2$

are strict monomorphisms. ▮

In the next definition we introduce a concatenation operator between trees and then we prove that it is a tensor product, i.e. an associative binary functor with unity.

DEFINITION 2.6. Given two trees, $t_1 = (X_1, \alpha_1, \beta_1)$ and $t_2 = (X_2, \alpha_2, \beta_2)$ sequential composition $\otimes$ is defined as follows (here $\cdot$ is used to denote string concatenation): $t_1 \otimes t_2 = \langle X, \alpha, \beta \rangle$, where

— $X = X_1 \times X_2$ (a run in t is a run of t_1 followed by a run of t_2);

— $\alpha(\langle x_1, x_2 \rangle) = \alpha_1(x_1) \cdot \alpha_2(x_2)$ (the labels of runs in t are obtained by concatenating those of the arguments);

— $\beta(\langle x_1, x_2 \rangle, \langle y_1, y_2 \rangle)$

$$\begin{aligned} &= \beta_1(x_1, y_1) \cdot \beta_2(x_2, y_2), && \text{if } x_1 = y_1, \\ &= \beta_1(x_1, y_1), && \text{otherwise} \end{aligned}$$

(the agreement between the second components of two composite runs is considered only if the runs have a common initial part).

PROPOSITION 2.7. *Sequential composition* $\otimes$ *is a tensor product with object unit tree* $\mathbf{1} = (\bullet, \alpha(\bullet) = \varepsilon, \beta(\bullet, \bullet) = \varepsilon)$ *and* $\mathbf{T}$ *is monoidal w.r.t.* $\otimes$.

TABLE 6

Axioms for Finite Nondeterministic Regular Expressions

$X+Y=Y+X$	(C1)
$(X+Y)+Z=X+(Y+Z)$	(C2)
$X+0=X$	(C3)
$(X \bullet Y) \bullet Z=X \bullet (Y \bullet Z)$	(S1)
$X \bullet 1=X$	(S2)
$1 \bullet X=X$	(S3)
$X \bullet 0=0$	(S4)
$0 \bullet X=0$	(S5)
$(X+Y) \bullet Z=(X \bullet Z)+(Y \bullet Z)$	(RD)

Terms of **PL** can be interpreted as trees in the category **T** by means of a function $\mathscr{T}$ defined by induction on the structure of terms.

DEFINITION 2.8 (Denotational semantics). An algebraic interpretation of finite **PL** terms is obtained by associating to them a tree in **T** via function $\mathscr{T}$:

— $\mathscr{T}[\![0]\!]=\mathbf{0}$,
— $\mathscr{T}[\![1]\!]=\mathbf{1}$,
— $\mathscr{T}[\![a]\!]=(x, \alpha(x)=a, \beta(x, x)=a)$,
— $\mathscr{T}[\![P+Q]\!]=\mathscr{T}[\![P]\!] \oplus \mathscr{T}[\![Q]\!]$,
— $\mathscr{T}[\![P \bullet Q]\!]=\mathscr{T}[\![P]\!] \otimes \mathscr{T}[\![Q]\!]$.

If we restrict ourselves to the subcategory $\mathbf{T}^{\mathrm{fin}}$ of finite trees, we can prove that $Tree^{\mathrm{fin}}$, the set of its objects, is the free model for the axioms of Table 6, i.e. those of Table 1 without those of Table 3.

For proving the main theorem of this section, we need a lemma that allows us to reduce **PL**-terms to standard forms.

DEFINITION 2.9 (Normal forms). A normal form is either 0 or a term of the form

$$\left(\sum_{i \in I} a_i+\sum_{j \in J} a_j \bullet n_j\right)+\sum_{k \in K} P_k,$$

where $P_k=1$ for all k and for all j we have that n_j is a normal form different from 0 and 1.

LEMMA 2.10 (Reduction to normal forms). *Every finite* **PL** *term P is provably equal, via the laws of Table* 6, *to a normal form $nf(P)$.*

Proof. The proof proceeds by induction on the depth of terms, defined by

$$\begin{aligned} \mathrm{depth}(0) &= 0 \\ \mathrm{depth}(1)=\mathrm{depth}(a) &= 1 \\ \mathrm{depth}(P+Q) &= \max\{\mathrm{depth}(P), \mathrm{depth}(Q)\} \\ \mathrm{depth}(P \bullet Q) &= \mathrm{depth}(P)+\mathrm{depth}(Q). \end{aligned}$$

Let us assume that the claim holds for terms P with $\text{depth}(P) < n$. We prove it for terms P of depth n. The proof proceeds by (inner) induction on the syntactic structure of terms.

(1) In case $P = 0$, $P = a$ and $P = 1$ the claim follows trivially: $nf(P) = P$.

(2) $P = P_1 + P_2$. By structural induction there exist two normal forms $nf(P_1)$ and $nf(P_2)$ such that $P_1 = nf(P_1)$ and $P_2 = nf(P_2)$. To obtain $nf(P_1 + P_2)$, we perform a case analysis:

(a) $nf(P_1) = 0$ and $nf(P_2) \neq 0$. Then $P_1 + P_2 = nf(P_1) + nf(P_2) = 0 + nf(P_2) = nf(P_2)$ by axiom (C3) and $nf(P_2)$ is a normal form.

(b) $nf(P_1) \neq 0$ and $nf(P_2) = 0$. Then $P_1 + P_2 = nf(P_1) + nf(P_2) = nf(P_1) + 0 = nf(P_1)$ by axiom (C3) and $nf(P_1)$ is a normal form.

(c) $nf(P_1) = 0$ and $nf(P_2) = 0$. Then $P_1 + P_2 = nf(P_1) + nf(P_2) = 0 + 0 = 0$ by axiom (C3) and 0 is a normal form.

(d) $nf(P_1) \neq 0$ and $nf(P_2) \neq 0$. Then $P_1 + P_2 = nf(P_1) + nf(P_2)$ is a normal form by axioms (C1) and (C2).

(3) $P = P_1 \bullet P_2$. By structural induction there are two normal forms $nf(P_1)$ and $nf(P_2)$ such that $P_1 = nf(P_1)$ and $P_2 = nf(P_2)$. We distinguish now a few cases to obtain a normal form $nf(P_1 \bullet P_2)$:

(a) $nf(P_1) = 0$ and $nf(P_2) \neq 0$. Then $P_1 \bullet P_2 = nf(P_1) \bullet nf(P_2) = 0 \bullet nf(P_2) = 0$ by axiom (S5) and hence $nf(P_1 \bullet P_2) = 0$.

(b) $nf(P_1) \neq 0$ and $nf(P_2) = 0$. Symmetric of a, use (S4) instead of (S5).

(c) $nf(P_1) = 0$ and $nf(P_2) = 0$. Then $P_1 \bullet P_2 = nf(P_1) \bullet nf(P_2) = 0 \bullet 0 = 0$ by axiom (S4) and, hence, $nf(P_1 \bullet P_2) = 0$.

(d) $nf(P_1) = 1$ and $nf(P_2) \neq 1$. Then $P_1 \bullet P_2 = nf(P_1) \bullet nf(P_2) = 1 \bullet nf(P_2) = nf(P_2)$ by axiom (S3) and $nf(P_1 \bullet P_2) = nf(P_2)$.

(e) $nf(P_1) \neq 1$ and $nf(P_2) = 1$. Symmetric of d, use (S2) instead of (S3).

(f) $nf(P_1) = 1$ and $nf(P_2) = 1$. Then $P_1 \bullet P_2 = nf(P_1) \bullet nf(P_2) = 1 \bullet 1 = 1$ by axiom (S2) and hence, $nf(P_1 \bullet P_2) = 1$.

(g) $nf(P_1) \neq 0, 1$ and $nf(P_2) \neq 0, 1$. Then

$$
\begin{aligned}
&P_1 \bullet P_2 \\
&\quad = nf(P_1) \bullet nf(P_2) \\
&\quad = \left(\left(\sum_{i \in I} a_i + \sum_{j \in J} a_j \bullet n_j \right) + \sum_{k \in K} P_k \{ P_k = 1 \} \right) \bullet nf(P_2) \\
&\qquad \text{by axiom (RD),} \\
&\quad = \left(\left(\sum_{i \in I} a_i \right) \bullet nf(P_2) + \left(\sum_{j \in J} a_j \bullet n_j \right) \bullet nf(P_2) \right) + \left(\sum_{k \in K} P_k \{ P_k = 1 \} \right) \bullet nf(P_2) \\
&\qquad \text{by axiom (RD),}
\end{aligned}
$$

$$= \left(\sum_{i \in I} a_i \bullet nf(P_2) + \sum_{j \in J} (a_j \bullet n_j) \bullet nf(P_2) \right) + \sum_{k \in K} P_k\{P_k = 1\} \bullet nf(P_2)$$

by axioms (S1), (S3),

$$= \left(\sum_{i \in I} a_i \bullet nf(P_2) + \sum_{j \in J} a_j \bullet (n_j \bullet nf(P_2)) \right) + \sum_{k \in K} nf(P_2)$$

by inductive hypothesis ($\text{depth}(n_j \bullet nf(P_2)) < n$),

$$= \left(\sum_{i \in I} a_i \bullet nf(P_2) + \sum_{j \in J} a_j \bullet nf(n_j \bullet nf(P_2)) \right) + \sum_{k \in K} nf(P_2)$$

Now, to prove the claim, it suffices to notice that the final term is a sum of normal forms and can thus be reduced to a normal form (see item 2 above). ∎

THEOREM 2.11 ($Tree^{\text{fin}}$ is the free model of (*-free) nondeterministic regular expressions.) *($Tree^{\text{fin}}$, $\oplus$, $\otimes$, $\mathbf{0}$, $\mathbf{1}$) is equivalent to the free model induced by the laws of Table* 6.

Proof. Given a term in the language **PL**, we can associate with it a unique (up to isomorphism) tree. To show that every tree has a unique description in the term algebra quotiented by our laws, we describe the normal form corresponding to each tree and prove that two different normal forms give rise to nonisomorphic trees.

We start by showing that any tree can be seen as a normal form; i.e., given a tree $t = (X, \alpha, \beta)$, there exists a normal form n such that $\mathscr{T}[\![n]\!] = t$.

The simplest case is when $t = (X, \alpha, \beta)$ with $X = \varnothing$. Then t coincides with $\mathscr{T}[\![0]\!]$.

If $X \neq \varnothing$ we proceed by induction on $d(t)$, the depth of t, defined here as the length of run $x \in X$ with maximum extent.

Assume $d(t) = 0$. In this case $\forall x \in X$, $\alpha(x) = \varepsilon$ and $\mathscr{T}[\![\sum_{k \in [1 \cdots |X|]} P_k\{P_k = 1\}]\!] = t$. Assume now $d(t) > 0$. For every $a \in A$, consider set $X_a = \{X_a^1, ..., X_a^k\}$, where $X_a^j = \{x_i \mid \beta(x_j, x_i) = a\}$. Intuitively, set X_a^j contains all runs $x_i \in X$ that have agreement greater or equal than a. Clearly, $X_a^j = X_a^i$ for every $x_i \in X_a^j$. Every $X_a^j \in X_a$ induces a tree $t_a^j = (X_a^j, \alpha_a^j, \beta_a^j)$, where α_a^j and β_a^j are the obvious restrictions of α and β over set of runs X_a^j. We will show that for every $X_a^j \in X_a$ there exists a normal form n_a^j such that $\mathscr{T}[\![n_a^j]\!] = t_a^j$ so that $n_a = \sum_{j \in [1 \cdots k]} n_a^j$ is a normal form for tree $t_a = (X_a^1 \cup \cdots \cup X_a^k, \alpha_a, \beta_a)$ with α_a and β_a the restrictions of α and β over set of runs $X_a^1 \cup \cdots \cup X_a^k$. Since there is a finite number of $a \in A$ such that $X_a \neq \varnothing$, it follows that $t = \mathscr{T}[\![\sum_{a \in A,\, X_a \neq \varnothing} n_a]\!]$. Thus, take $X_a^j \in X_a$ and consider tree $t_a^j = (X_a^j, \alpha_a^j, \beta_a^j)$.

If $d(t) = 1$ then $\forall x \in X_a^j$, $\alpha(x) = a$ and we have either $t_a^j = \mathscr{T}[\![a]\!]$ if $|X_a^j| = 1$ or $t_a^j = \mathscr{T}[\![a \bullet \sum_{k \in |X_a^j|} P_k\{P_k = 1\}]\!]$ if $|X_a^j| > 1$.

Assume now, by induction hypothesis, that for every t_a^j such that $1 < d(t_a^j) \leqslant n$ there exists a normal form n_a^j such that $\mathscr{T}[\![n_a^j]\!] = t_a^j$. We prove that the statement for any t_a^j with $d(t_a^j) = n + 1$. In this case X_a^j induces a tree $t_a'^j = (X_a^j, \alpha_a'^j, \beta_a'^j)$, where $\alpha_a'^j$ and $\beta_a'^j$ are defined by $\alpha_a'^j(x_a^j) = w$ if $\alpha(x_a^j) = aw$ and $\beta_a'^j(x_a^i, x_a^j) = w$ if $\beta(x_a^i, x_a^j,) = aw$ for $x_a^i, x_a^j \in X_a^j$. Clearly, $d(t_a^j) > d(t_a'^j)$; thus by induction hypothesis there exists a normal form $n_a'^j$ such that $\mathscr{T}[\![n_a'^j]\!] = t_a'^j$ and, hence, $\mathscr{T}[\![a \bullet n_a'^j]\!] = t_a^j$.

It remains to be proven that two different normal forms give rise to non-isomorphic trees. But this immediately follows by an inspection of the normal forms. Indeed if they differ over summands P_k with $P_k = 1$ then they have a different number of 1-summands and, hence, the corresponding trees cannot be isomorphic. Similarly, if they differ over summands a. For summands of the form $a \bullet n$ the claim follows from an inductive reasoning. ▌

3. OPERATIONAL AND OBSERVATIONAL MODELS

Here we provide an observational account of finite nondeterministic regular expressions, by interpreting them as equivalence classes of labelled transition systems. The proposed equivalence relies on the same recursive pattern of bisimulation but takes into account also the number of equivalent states that are reachable from a given one.

DEFINITION 3.1. A labelled transition system is a triple $\langle Z, L, T \rangle$, where

— Z is a set of states,

— L is a set of labels

— T is a transition relation; $T \subseteq Z \times L \times Z$.

The elements of T will often represented as $q \xrightarrow{l} q'$, rather than as triples; thus, we shall write $z \xrightarrow{l} z'$, instead of $\langle z, l, z' \rangle \in T$.

In our case, states are terms of **PL** (as defined in the Introduction) and labels are pairs $\langle \mu, u \rangle$ with $\mu \in A \cup \{1\}$ and u a term, called *choice sequence*, generated by

$$u ::= \varepsilon \mid lu \mid ru \quad \text{with } l, r \text{ tags.}$$

The transition relation relies on the predicate defined in Table 7 and is defined in Table 8. For those familiar with the operational semantic of process algebras, we would like to remark that 1-actions do not play the same role of invisible τ-actions. They simply stand for successfully terminated states.

We have two kinds of transitions:

- $P \xrightarrow{\langle a, u \rangle} P'$: P performs an action a, possibly preceded by 1-actions with choice sequence u.
- $P \xrightarrow{\langle 1, u \rangle} 1$: P performs 1-actions to reach process 1 with choice sequence u.

TABLE 7

Active Predicate

$\text{active}(1)$
$\text{active}(a)$
$\text{active}(P) \vee \text{active}(Q) \Rightarrow \text{active}(P + Q)$
$\text{active}(P) \wedge \text{active}(Q) \Rightarrow \text{active}(P \bullet Q)$

TABLE 8

Operational Semantics for **PL**

(Tic)	$\dfrac{}{1 \xrightarrow{\langle 1, \varepsilon \rangle} 1}$	(Atom)	$\dfrac{}{a \xrightarrow{\langle a, \varepsilon \rangle} 1}$
(Sum_1)	$\dfrac{P \xrightarrow{\langle \mu, u \rangle} P'}{P + Q \xrightarrow{\langle \mu, lu \rangle} P'}$	(Sum_2)	$\dfrac{Q \xrightarrow{\langle \mu, u \rangle} Q'}{P + Q \xrightarrow{\langle \mu, ru \rangle} Q'}$
(Seq_1)	$\dfrac{P \xrightarrow{\langle a, u \rangle} P', \mathrm{active}(Q)}{P \bullet Q \xrightarrow{\langle a, u \rangle} P' \bullet Q}$	(Seq_2)	$\dfrac{P \xrightarrow{\langle 1, u \rangle} 1, Q \xrightarrow{\langle \mu, u' \rangle} Q'}{P \bullet Q \xrightarrow{\langle \mu, uu' \rangle} Q'}$

These transitions are atomic; they cannot be interrupted and they keep no track of intermediate states. In both cases, u is used to keep information about the possible nondeterministic structure of P and will permit distinguishing those transitions of P whose action label and target state have the same name. Thus for $a + a$, it is possible to record that it can perform two different a actions: $a + a \xrightarrow{\langle a, l \rangle} 1$ and $a + a \xrightarrow{\langle a, r \rangle} 1$. Without l and r, we would have only the $a + a \xrightarrow{a} 1$ transition.

The predicate *active* over **PL** processes used in Seq_1 allows us to detect empty processes and to avoid performing actions leading to deadlock.

The rules of Table 8 should be self-explanatory. We only comment on those for $+$ and $\bullet$.

The rule for $P + Q$ says that if P can perform $\langle \mu, u \rangle$ to become P' then $P + Q$ can perform $\langle \mu, lu \rangle$ to become P', where l records that action μ has been performed by the left alternative. The right alternative is dealt with symmetrically. (Seq_1) mimics sequential composition of P and Q; it states that if P can perform $\langle \mu, r \rangle$ then $P \bullet Q$ can evolve with the same label to $P' \bullet Q$. The premise active(Q) of the inference rule ensures that Q can successfully terminate. Note that active(Q) in (Seq_1) could be replaced by $\exists Q' \cdot Q \xrightarrow{\langle \mu, u' \rangle} Q'$, that is by requiring Q to perform any transition. This choice, however, would require a "look ahead" that would be heavy when mechanically checking successful termination of a process. We can, instead, statically check whether a process eventually reaches a deadlock state.

In order to abstract from choice sequences while keeping information about the alternatives a process has for performing a specific action, we introduce a new transition relation that associates to every pair $\langle P \in \mathbf{PL}, \mu \in Act \cup \{1\} \rangle$, a multiset M, representing all processes that are target of $\langle \mu, u \rangle$-transitions from P. It is defined as the least relation such that

$$P \xrightarrow{\mu} \{\!|P' \mid \exists u \cdot P \xrightarrow{\langle \mu, u \rangle} P'|\!\}.$$

Thus, we have

- $a + a \xrightarrow{a} \{\!|1, 1|\!\}$ because
  - $a + a \xrightarrow{\langle a, l \rangle} 1$
  - $a + a \xrightarrow{\langle a, r \rangle} 1$;

- $(1+1)\bullet(a+a) \xrightarrow{a} \{\!|1, 1, 1, 1, |\!\}$ because
  - $(1+1)\bullet(a+a) \xrightarrow{\langle a, ll\rangle} 1$
  - $(1+1)\bullet(a+a) \xrightarrow{\langle a, lr\rangle} 1$
  - $(1+1)\bullet(a+a) \xrightarrow{\langle a, rl\rangle} 1$
  - $(1+1)\bullet(a+a) \xrightarrow{\langle a, rr\rangle} 1.$

We also have that $1+1 \xrightarrow{1} \{\!|1, 1|\!\}$ while $1 \xrightarrow{1} \{\!|1|\!\}$. Here we would like to remark that, with the proposed semantics, we can count also 1-transitions. If we did not have this possibility, we would have been forced to identify $X+X$ and X by $(1+1)\bullet X$ and $1\bullet X$.

3.1. Resource Simulation and Resource Bisimulation

Very often descriptions via labelled transition systems turn out to be too concrete. To abstract from "irrelevant" details and for relating different descriptions of the same system, the notions of behavioral relations (*equivalences* or *preorders*) are often used. Different opinions about the relevant features of concurrent systems to be taken into account and, thus, about the aspects that can be ignored have led to a number of behavioral relations. A few of these notions are based on the notions of *bisimulation*. Intuitively, two systems are bisimulation equivalent whenever they can perform the same sequences of actions to reach (via them) bisimulation equivalent states.

In this subsection, we will introduce two new bimulation-based relations that aim at identifying only those systems that have exactly the same behaviour and, thus, differ only for their syntactic structure. We will introduce an equivalence relation, *resource bisimulation*, that relates only those terms whose unfolding, via the operational semantics, gives rise to isomorphic labelled trees. We will also introduce a preorder, *resource simulation*, that "captures" the notion of tree embedding. A preliminary report on the result of this section appeared as [13].

The transition relation $\xrightarrow{\mu}$, introduced above, is the basis for defining *resource simulation* and *resource bisimulation*.

DEFINITION 3.2 (Resource simulation and resource bisimulation). 1a. A relation $\mathfrak{R} \subseteq \mathbf{PL} \times \mathbf{PL}$ is a *r-simulation* if for each $\langle P, Q\rangle \in \mathfrak{R}$ and for each $\mu \in A \cup \{1\}$: $P \xrightarrow{\mu} M$ implies $Q \xrightarrow{\mu} M'$ and $\exists f$ injective: $M \to M'$, s.t. $\forall P' \in M$, $\langle P', f(P')\rangle \in \mathfrak{R}$;

1b. P and Q are *r-similar* ($P \lhd_r Q$) if there exists a *r-simulation* $\mathfrak{R}$ containing $\langle P, Q\rangle$.

2a. A relation $\mathfrak{R} \subseteq \mathbf{PL} \times \mathbf{PL}$ is a *r-bisimulation* if for each $\langle P, Q\rangle \in \mathfrak{R}$, and for each $\mu \in A \cup \{1\}$:

— $P \xrightarrow{\mu} M$ implies $Q \xrightarrow{\mu} M'$ and $\exists f$ injective: $M \to M'$, s.t. $\forall P' \in M$, $\langle P', f(P')\rangle \in \mathfrak{R}$;

— $Q \xrightarrow{\mu} M'$ implies $P \xrightarrow{\mu} M$ and $\exists g$ injective: $M' \to M$, s.t. $\forall Q' \in M'$, $\langle Q', g(Q')\rangle \in \mathfrak{R}$;

2b. P and Q are *r-bisimilar* ($P \sim_r Q$), if there exists a *r-bisimulation* $\mathfrak{R}$ containing $\langle P, Q\rangle$.

The above definitions should be self-explanatory. We want simply to remark that the injection $f: M \to M'$ is used to ensure that different (indexed) processes in M are simulated by different (indexed) processes in M'.[6] Thus r-bisimilarity requires the cardinality of M be less or equal to the cardinality of M'.

With standard techniques it is possible to show that $\sim_r$ is an equivalence relation and it is preserved by nondeterministic composition and sequential composition. It is not difficult to check that $a \lhd_r a + a$, $a \not\lhd_r a + a$, $a + b \sim_r b + a$, and $(1 + 1) \bullet a \sim_r a + a$.

The following propositions provide soundness and completeness results for the axiomatization of r-simulation and r-bisimulation over finite **PL** terms.

First, we will prove that the laws of Table 6, an inequational rule, $X \leqslant X + Y$, that captures the essence of resource simulation, and two inference rules stating monotonicity of $\leqslant$ with respect $\bullet$ and $+$ (see Table 4) soundly and completely axiomatize r-simulation over finite **PL**. Clearly, $X = Y$ in Table 6 has to be intended now as $X \leqslant Y$ and $Y \leqslant X$.

Then, we will prove that the laws of Table 1, without of those in Table 3, are sufficient to axiomatize r-bisimulation over finite **PL**.

First we establish a congruence result for our preorder.

PROPOSITION 3.3. *Resource simulation is preserved by all* **PL** *operators.*

Proof. We just prove the statement for $\bullet$ which is the most involved case. The proof for $+$ is simpler. We prove that given $X \lhd_r Y$ and $R \lhd_r S$; we have $X \bullet R \lhd_r Y \bullet S$. To prove this result, we show that

$$\mathfrak{R} = \{(X \bullet R,\ Y \bullet S) \mid X \lhd_r Y \text{ and } R \lhd_r S\} \cup \mathfrak{R}_1 \cup \mathfrak{R}_2,$$

where $\mathfrak{R}_1$ and $\mathfrak{R}_2$ are the simulation relations used to establish $X \lhd_r Y$ and $R \lhd_r S$, is a resource simulation. Assume $(X \bullet R, Y \bullet S) \in \mathfrak{R}$ and $X \lhd_r Y$, $R \lhd_r S$. Consider $X \bullet R \xrightarrow{\mu} M$. Then we prove that $Y \bullet S \xrightarrow{\mu} M'$ and $\exists f$ injective: $M \to M'$, s.t. $\forall P \in M$, $\langle P, f(P) \rangle \in \mathfrak{R}$. We distinguish two cases depending on $\mu = 1$ or $\mu = a$:

— $\mu = 1$. Then $M = \{\!|P \mid P = 1|\!\}$. If $|M| = k$, then there are k transitions of the form $X \bullet R \xrightarrow{\langle 1, u_i \rangle} 1$ for $i \in [1 \cdots k]$. By an inspection of the rules in Table 8 these transitions are of the form $X \xrightarrow{\langle 1, u' \rangle} 1$ and $R \xrightarrow{\langle 1, u'' \rangle} 1$ with $u_i = u'_i u''_i$. Now, since $X \lhd_r Y$, for every $X \xrightarrow{\langle 1, u'_i \rangle} 1$ there exists a different $Y \xrightarrow{\langle 1, v'_i \rangle} 1$ and, since $R \lhd_r S$, for every $R \xrightarrow{\langle 1, u''_i \rangle} 1$, there exists a different $S \xrightarrow{\langle 1, v''_i \rangle} 1$. Thus, for every $X \bullet R \xrightarrow{\langle 1, u'_i u''_i \rangle} 1$ there exists a different $Y \bullet S \xrightarrow{\langle 1, v'_i v''_i \rangle} 1$ in M'. It follows that there exists an injection from M to M'.

— $\mu = a$ and consider $X \bullet R \xrightarrow{\mu} M$. Note that every process in M can be either of the form $X' \bullet R$ or of the form R'. In particular, the former processes are target of transitions $X \bullet R \xrightarrow{\langle a, u \rangle} X' \bullet R$ if $X \xrightarrow{\langle a, u \rangle} X'$, while the latter processes are target states of transitions of the form $X \bullet R \xrightarrow{\langle a, u \rangle} R'$ if $X \xrightarrow{\langle 1, u' \rangle} 1$, $R \xrightarrow{\langle a, u'' \rangle} R'$ and $u = u'u''$. Clearly, since $X \lhd_r Y$, every transition $X \bullet R \xrightarrow{\langle a, u \rangle} X' \bullet R$ with $X \xrightarrow{\langle a, u \rangle} X'$ has a different transition $Y \bullet S \xrightarrow{\langle a, v \rangle} Y' \bullet S$ with $Y \xrightarrow{\langle a, v \rangle} Y'$ and

[6] Since a multiset can be seen as a set of indexed elements, an injection between multisets will be seen just as an ordinary injection between sets.

$(X' \bullet R, Y' \bullet S) \in \mathfrak{R}$. Similarly, since $X \lhd_r Y$ and $R \lhd_r S$, every transition $X \bullet R \xrightarrow{\langle a, u \rangle} R'$ with $X \xrightarrow{\langle 1, u' \rangle} 1$ and $R \xrightarrow{\langle a, u'' \rangle} R'$ and $u = u'u''$ has a different transition $Y \bullet S \xrightarrow{\langle a, v \rangle} S'$ with $Y \xrightarrow{\langle 1, v' \rangle} 1$, $S \xrightarrow{\langle a, v'' \rangle} S'$, $v = v'v''$, and $(R', S') \in \mathfrak{R}_2$. ▮

PROPOSITION 3.4 (Completeness for r-simulation). *The laws of Table* 6 *and Table* 4 *soundly and completely axiomatize r-simulation over finite* **PL**.

Proof. It is possible to show that the following statements hold, once we let $\leqslant$ denote the preorder over finite **PL** terms induced by the rules of Table 4:

Soundness. For all finite **PL** terms P and Q, $P \leqslant Q$ implies $P \lhd_r Q$. This can be proved by showing the appropriate resource simulations. Here we just prove soundness of axioms (S2) and (RD). All the other axioms can be proven similarly.

To show that $X \bullet 1 = X$, we prove that both $X \bullet 1 \lhd_r X$ and $X \lhd_r X \bullet 1$ hold. These two statements follow the fact that relations

$$\mathfrak{R} = \{(X \bullet 1, X) \mid X \in \mathbf{PL}\} \cup \{(1, 1)\}; \quad \mathfrak{R}' = \{(X, X \bullet 1) \mid X \in \mathbf{PL}\} \cup \{(1, 1)\}$$

are resource simulations. Indeed,

(1) $X \bullet 1 \xrightarrow{\langle 1, \mu \rangle} 1$ iff $X \xrightarrow{\langle 1, \mu \rangle} 1$

(2) $X \bullet 1 \xrightarrow{\langle a, u \rangle}$ iff $X \xrightarrow{\langle a, u \rangle} X'$

can be easily proven by a simple inspection of the operational rules.

To prove that $(X + Y) \bullet Z = (X \bullet Z) + (Y \bullet Z)$, we prove that $(X + Y) \bullet Z \lhd_r (X \bullet Z) + (Y \bullet Z)$ and that $(X \bullet Z) + (Y \bullet Z) \lhd_r (X + Y) \bullet Z$. Again this follows by showing that relations

$$\mathfrak{R} = \{((X + Y) \bullet Z, (X \bullet Z) + (Y \bullet Z)) \mid X, Y, Z \in \mathbf{PL}\} \cup id;$$

$$\mathfrak{R}' = \{((X \bullet Z) + (Y \bullet Z), (X + Y) \bullet Z) \mid X, Y, Z \in \mathbf{PL}\} \cup id$$

are resource simulations. This can be proven via the items that are direct consequences of the operational semantics:

(1) $(X + Y) \bullet Z \xrightarrow{\langle \mu, lu \rangle} R$ iff $(X \bullet Z) + (Y \bullet Z) \xrightarrow{\langle \mu, lu \rangle} R$;

(2) $(X + Y) \bullet Z \xrightarrow{\langle \mu, ru \rangle} R$ iff $(X \bullet Z) + (Y \bullet Z) \xrightarrow{\langle \mu, ru \rangle} R$.

Completeness. For all normal forms n_1 and n_2, $n_1 \lhd_r n_2$ implies $n_1 \leqslant n_2$. This result can be proven as in [27] by using axioms in Table 6. The proof proceeds by induction on $\text{depth}(n_1) + \text{depth}(n_2)$. Assume the statement for $\text{depth}(n_1) + \text{depth}(n_2) < n$ and prove it for $\text{depth}(n_1) + \text{depth}(n_2) = n$. Let n_1 and n_2 be

$$n_1 = \sum_{i \in I} a_i + \sum_{j \in J} a_j \bullet n_j + \sum_{k \in K} \{P_k = 1\};$$

$$n_2 = \sum_{l \in L} b_l + \sum_{m \in M} b_m \bullet s_m + \sum_{n \in N} \{P_n = 1\}.$$

Moreover, by hypothesis we have $n_1 \lhd_r n_2$. Let $\{a_1, a_2, ..., a_n\}$ be the set of initial actions appearing in n_1; that is, $\{a_1, a_2, ..., a_n\} = \{a_i \mid i \in I\} \cup \{a_j \mid j \in J\}$. Then for proper subsets of I, J, L, and M,

$$n_1 = \left(\sum_{1_i \in I_1} \{a_{1_i} = a_1\} + \sum_{1_j \in J_1} a_1 \bullet n_{1_j} \right) + \cdots + \left(\sum_{n_i \in I_n} \{a_{n_i} = a_n\} + \sum_{n_j \in J_n} a_n \bullet n_{n_j} \right) + \sum_{k \in K} \{P_k = 1\};$$

$$n_2 = \left(\sum_{1_l \in L_1} \{b_{1_l} = a_1\} + \sum_{1_m \in M_1} a_1 \bullet s_{1_m} \right) + \cdots + \left(\sum_{n_l \in L_n} \{b_{n_l} = a_n\} + \sum_{n_m \in L_n} a_n \bullet s_{n_m} \right) + \sum_{n \in N} \{P_n = 1\} + r,$$

where r is a normal form that does not have initial $\{a_1, a_2, ..., a_n\}$-actions and initial 1-actions. By the definition of resource simulation, it is easy to see that

(i) for every $o \in [1 \cdots n]$,

$$\left(\sum_{o_i \in I_o} \{a_{o_i} = a_o\} + \sum_{o_j \in J_o} a_o \bullet n_{o_j} \right) \lhd_r \left(\sum_{o_l \in L_o} \{b_{o_l} = a_o\} + \sum_{o_m \in M_o} a_o \bullet s_{o_m} \right)$$

(ii) and summations of 1's are related:

$$\sum_{k \in K} \{P_k = 1\} \lhd_r \sum_{n \in N} \{P_n = 1\}.$$

Thus, if we are able to prove the main result for (i) and (ii), then by axioms (R2) and (PR) (useful to deal with r in n_2) we are also able to prove that $n_1 \leqslant n_2$. We just prove (i) because (ii) is clearly simple. Again we prove (i) when $o = 1$; all other cases follow similarly. Assume

$$\left(\sum_{1_i \in I_1} \{a_{1_i} = a_1\} + \sum_{1_j \in J_1} a_1 \bullet n_{1_j} \right) \lhd_r \left(\sum_{1_l \in L_1} \{b_{1_l} = a_1\} + \sum_{1_m \in M_1} a_1 \bullet s_{1_m} \right). \tag{1}$$

Then

$$\sum_{1_i \in I_1} \{a_{1_i} = a_1\} + \sum_{1_j \in J_1} a_1 \bullet n_{1_j} \xrightarrow{a_1} \{\!\{ \underbrace{1, ..., 1}_{|I_1| \text{ times}}, n_{1_{j_1}}, ..., n_{1_{j_{|J_1|}}} \}\!\} = M,$$

where every $n_{1_{j_i}}$ is different from 1 by definition of the normal form. Similarly,

$$\sum_{1_l \in L_1} \{b_{1_l} = a_1\} + \sum_{1_m \in M_1} a_1 \bullet s_{1_m} \xrightarrow{a_1} \{\!\{ \underbrace{1, ..., 1}_{|L_1| \text{ times}}, s_{1_{m_1}}, ..., s_{1_{m_{|M_1|}}} \}\!\} = M'.$$

By hypothesis there exists an injection $f: M \to M'$ such that $\forall P \in M$, $P \lhd_r f(P)$. We have to distinguish three cases:

(a) $P = 1$ and $f(P) = 1$. Then we have $1 \leqslant 1$ by axiom (R1) is $a_1 \bullet 1 \leqslant a_1 \bullet 1$. Finally, by axiom (S2) is $a_1 \leqslant a_1$.

(b) $P=1$ and $f(P)=s_{1_{m_k}}$ for some k. Then necessarily $s_{1_{m_k}}$ has a summand 1 and, hence, by axiom (PR) (and (C1), (C2)) it follows that $1 \leqslant s_{1_{m_k}}$. Thus, also, $a_1 \bullet 1 \leqslant a_1 \bullet s_{1_{m_k}}$ and by axiom (S2) is $a_1 \leqslant a_1 \bullet s_{1_{m_k}}$.

(c) $P=n_{1_{j_k}}$ for some k. Then $f(P)$ cannot be 1. Let $f(P)=s_{1_{m_{k'}}}$ for some k'. Here we can apply induction hypothesis to prove that $n_{1_{j_k}} \leqslant s_{1_{m_{k'}}}$. By axiom $R1$, $a_1 \bullet n_{1_{j_k}} \leqslant a_1 \bullet s_{1_{m_{k'}}}$.

Items (a), (b), and (c) above show that every summand of $\sum_{1_i \in I_1} \{a_{1_i}=a_1\} + \sum_{1_j \in J_1} a_1 \bullet n_{1_j}$ is provably $\leqslant$ to a summand of $\sum_{1_l \in L_1} \{b_{1_l}=a_1\} + \sum_{1_m \in M_1} a_1 \bullet s_{1_m}$. Thus, in order to complete the proof for (1) we just need to eventually apply axiom (PR) (note, indeed, that the latter summation may have more summands of the former one). ▌

Similarly, we can establish the corresponding result for r-bisimulation.

PROPOSITION 3.5 (Completeness for r-bisimulation). *The laws of Table* 6 *soundly and completely axiomatize r-bisimulation over finite* **PL**.

3.2. *A Preorder Whose Kernel is Resource Equivalence*

In this section we show that the kernel of $\lhd_r$ coincides with resource equivalence. This result is new for simulation-like semantics; for example, it does not hold for the classical simulation preorder of [27, 28]. In that case we have that bisimulation cannot be obtained as a double simulation.[7] The usefulness of this coincidence result is twofold. First of all, it permits concentrating just on the preorder and to obtain as corollary many results (like the last theorem of the previous section) for the equivalence. Second, it permits using this behavioral relation for stepwise refinements of systems implementation.

In order to establish this property, we show that the set of pairs (P, Q) of **PL** processes such that $P \lhd_r Q$ and $Q \lhd_r P$ is a resource bisimulation. This will be an immediate consequence of the following two lemmas. They permit us to conclude that, given two sets of **PL** processes, S and S', and two injections $f: S \to S'$, $g: S' \to S$ such that $\forall s \in S$, $s \lhd_r f(s)$ and $\forall s' \in S'$, $s' \lhd_r g(s')$, then for each $s \in S$ we have $s \lhd_r f(s)$ and $f(s) \lhd_r s$ ($s' \lhd_r g(s')$ and $g(s') \lhd_r s'$), and similarly for s'.

LEMMA 3.6. *Let* $S=\{P_1, ..., P_n\}$ *and* $S'=\{Q_1, ..., Q_n\}$ *be two sets of* **PL** *processes and* $f: S \to S'$, $g: S' \to S$ *be two injections such that* $\forall P_i \in S$, $P_i \lhd_r f(P_i)$ *and* $\forall Q_i \in S'$, $Q_i \lhd_r g(Q_i)$. *Then for each* $P_{i_1} \in S$, $i_1 \in n[1 \cdots n]$ *there exists a chain* $P_{i_1} \lhd_r Q_{i_1} \lhd_r P_{i_2} \lhd_r Q_{i_2} \lhd_r \cdots \lhd_r Q_{i_m} \lhd_r P_{i_1}$ *for some* $m \in [1 \cdots n]$, $\{i_1, ..., i_m\} \subseteq [1 \cdots n]$, $P_{i_j} \in S$, $Q_{i_j} \in S'$ *and* $P_{i_j} \neq P_{i_k}$, $Q_{i_j} \neq Q_{i_k}$ *for each* $j \neq k$.

Proof. The existence of the chain follows by construction. Consider the chain $P_{i_1} \lhd_r Q_{i_1} \lhd_r \cdots \lhd_r P_{i_j} \lhd_r Q_{i_j} \lhd_r P_{i_{j+1}} \lhd_r \cdots \lhd_r Q_{i_m} \lhd_r P_{i_1}$ such that $Q_i = f(P_{i_j})$ and $P_{i_{j+1}} = g(Q_{i_j})$ for each $j \in [1 \cdots m-1]$. Clearly, it is $P_{i_j} \in S$ and $Q_{i_j} \in S'$. Prove that $P_{i_j} \neq P_{i_k}$ for each $j \neq k$. Consider in the construction the first index m such that there exists $2 \leqslant l < m$ for which $P_{i_l} = P_{i_m}$. But then, since g is an injection it is $Q_{i_{l-1}} = Q_{i_{m-1}}$, where $g(Q_{i_{l-1}}) = P_{i_j}$ and $g(Q_{i_{m-1}}) = P_{i_m}$, and since f is an injection it

[7] Consider terms $a \bullet b$ and $a + (a \bullet b)$. According to Park and Milner's definitions, we have that $a \bullet b$ simulates $a + (a \bullet b)$ and that $a + (a \bullet b)$ simulates $\tau \bullet b$, but they are not bisimilar.

is $P_{i_{l-1}} = P_{i_{m-1}}$, where $f(P_{i_{l-1}}) = Q_{i_{l-1}}$ and $f(P_{i_{m-1}}) = Q_{i_{m-1}}$. But this is impossible because m is the minimum index such that $P_{i_l} = P_{i_m}$. Thus, all P_{i_j} are different and for symmetrical arguments all Q_{i_j} are also different.

Finally we prove that the above chain ends with P_{i_1}. Suppose that $Q_{i_m} \lhd_r P_{i_1}$ for no index m; that is, $g(Q_{i_m}) = P_{i_1}$ for no index m. Since $\text{card}(S) = n$ and $\text{card}(S') = n$ (where $\text{card}(S)$ denotes the cardinality of finite set S) and every time we take a different process from S and S', we can assume that $P_{i_1} \lhd_r Q_{i_1} \lhd_r \cdots \lhd_r P_{i_n} \lhd_r Q_{i_n}$ and $g(Q_{i_n}) \neq P_{i_1}$. This is absurd. In fact g is an injection, thus $g(Q_{i_n}) \in S$, but if $g(Q_{i_n}) \neq P_{i_1}$ then $g(Q_{i_n}) = P_{i_j}$ for some $j \neq 1$. By construction, also, $g(Q_{i_{j-1}}) = P_{i_j}$ and $Q_{i_{j-1}} \neq Q_{i_n}$ because all Q_{i_j} in the chain are different. Now since g is an injection and it is not possible to have $g(Q_{i_{j-1}}) = g(Q_{i_n}) = P_{i_j}$ for $Q_{i_{j-1}} \neq Q_{i_n}$. Hence, $Q_{i_n} = P_{i_1}$. It follows that there exists an index m such that $g(Q_{i_m}) = P_{i_1}$. ∎

LEMMA 3.7. *Let $S = \{P_1, ..., P_n\}$ and $S' = \{Q_1, ..., Q_n\}$ be sets of* **PL** *processes and $f: S \to S'$, $g: S \to S$ be two injections such that $\forall P_i \in S$, $P_i \lhd_r f(P_i)$ and $\forall Q_i \in S'$, $Q_i \lhd_r g(Q_i)$. Then for each $P_i \in S$, $i \in [1 \cdots n]$, $P_i \lhd_r f(P_i)$ and $f(P_i) \lhd_r P_i$ (there exists an injection $r: S' \to S$ such that $r(f(P_i)) = P_i$).*

Proof. Let us suppose the existence of some P_{i_1}, $i_1 \in [1 \cdots n]$, such that $P_{i_1} \lhd_r f(P_{i_1})$ but $f(P_{i_1}) \not\lhd_r P_{i_1}$. By Lemma 3.6 there exists a chain starting by P_{i_1}, $P_{i_1} \lhd_r Q_{i_1} \lhd_r P_{i_2} \lhd_r Q_{i_2} \lhd_r \cdots \lhd_r Q_{i_m} \lhd_r P_{i_1}$ such that all P_{i_j} are different apart from P_{i_1}. Moreover, we know by hypothesis that $Q_{i_1} = f(P_{i_1})$ and $g(Q_{i_1}) = P_{i_2} \neq P_{i_1}$. Since $\lhd_r$ is a preorder it is transitive and then we have that $P_{i_1} \lhd_r P_{i_2} \lhd_r \cdots \lhd_r P_{i_1}$ and $Q_{i_1} \lhd_r Q_{i_2} \lhd_r \cdots \lhd_r Q_{i_m}$. Hence, also $Q_{i_1} \lhd_r Q_{i_m}$. Since $Q_{i_m} \lhd_r P_{i_1}$ follows by transitive property $Q_{i_1} \lhd_r P_{i_1}$. This contradicts the hypothesis because Q_{i_1} is just $f(P_{i_1})$. ∎

The coincidence between the kernel of resource simulation and resource equivalence is established by the following proposition.

PROPOSITION 3.8. *For processes P and Q, $P \sim_r Q$ iff $P \lhd_r Q$ and $Q \lhd_r P$.*

Proof. The case $P \sim_r Q$ implies $P \lhd_r Q$ and $Q \lhd_r P$ follows by definitions of $\sim_r$ and $\lhd_r$. To prove it vice versa we show that

$$\mathfrak{R} = \{\langle P_1, Q_1\rangle \mid P_1 \lhd_r Q_1 \text{ and } Q_1 \lhd_r P_1\}$$

is a r-bisimulation.

Consider a generic pair $\langle P_1, Q_1\rangle \in \mathfrak{R}$. Then it is $P_1 \lhd_r Q_1$ and $Q_1 \lhd_r P_1$. It follows that for each action $\mu \in A \cup \{1\}$ we have

— $P_1 \xrightarrow{\mu} M$ implies $Q_1 \xrightarrow{\mu} M'$ and there exists an injection $f: M \to M'$ such that $\forall P' \in M$ is $P' \lhd_r f(P')$;

— $Q_1 \xrightarrow{\mu} M'$ implies $P_1 \xrightarrow{\mu} M$ and there exists an injection $g: M' \to M$ such that $\forall Q' \in M'$ is $Q' \lhd_r g(Q')$.

To prove that $\mathfrak{R}$ is a resource bisimulation we have to show that both $\langle P', f(P')\rangle \in \mathfrak{R}$ and $\langle f(P'), P'\rangle \in \mathfrak{R}$. By Lemma 3.7 we have that $\forall P' \in M$, $P' \lhd_r f(P')$ and $f(P') \lhd_r P'$. Thus, $\langle P', f(P')\rangle$ and $\langle f(P'), P'\rangle$ are in $\mathfrak{R}$. Hence, $\mathfrak{R}$ is a resource bisimulation and, since $\langle P, Q\rangle \in \mathfrak{R}$, it follows that $P \sim_r Q$. ∎

4. COMPARING OBSERVATIONAL AND DENOTATIONAL SEMANTICS

In this section we compare observational and denotational semantics of regular expressions. To conclude that they coincide, it would be sufficient to observe that they are sound and complete with respect to the same set of axioms. We can, however, exhibit a more direct correspondence by showing that the tree obtained by unfolding the transition system associated to P, $LTS(P)$, is isomorphic to the tree obtained by interpreting process P via the interpretation function $\mathscr{T}$.

Clearly, the direct correspondence between the two semantics offers us different techniques for establishing properties of our formalism. For example, Proposition 4.2, Proposition 4.3, and Theorem 4.4 give an alternative way of proving Proposition 3.8.

DEFINITION 4.1. Let P be a term:

— $LTS(P)$ denotes the transition system associated to P according to the transition rules in Table 8;

— *run*(P) denotes the set of transition sequences, called runs or computations $P \xrightarrow{\langle \mu_1, u_1 \rangle} P_1 \xrightarrow{\langle \mu_2, u_2 \rangle} \cdots \xrightarrow{\langle \mu_{n-1}, u_{n-1} \rangle} P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1$ with $P_1, ..., P_{n-1} \neq 1$ performed by P;

— *forget* is a function from runs to sequences of actions. Given a run x, $P \xrightarrow{\langle \mu_1, u_1 \rangle} P_1 \xrightarrow{\langle \mu_2, u_2 \rangle} \cdots \xrightarrow{\langle \mu_{n-1}, u_{n-1} \rangle} P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} P'$, $\text{forget}(x) = \hat{\mu}_1 \hat{\mu}_2 \cdots \hat{\mu}_{n-1} \hat{\mu}_n$ extracts the sequence of performed actions, where we let $\hat{\mu}$ be equal to ε if $\mu = 1$, or otherwise $\hat{\mu} = \mu$.

— We define the weight of a transition system as the sum of the weights of its runs, that is, $\text{weight}(LTS(P)) = \sum_i \text{weight}(x_i)$, where $x_i \in \text{run}(P)$. The weight of a run x is defined by $\text{weight}(x) = 0$ if $\text{forget}(x) = \varepsilon$ and $\text{weight}(x) = n$ if $\text{forget}(x) = a_1 a_2 \cdots a_{n-1} a_n$. We also let $\text{weight}(P) = \text{weight}(LTS(P))$.

— The tree associated to $LTS(P)$ is $\text{Unf}(P) = \langle X, \alpha, \beta \rangle$, defined by $X = \{x \mid x \in \text{run}(P)\}$, $\alpha(x) = \text{forget}(x)$, $\beta(x, y) = \text{forget}(x \wedge y)$, where $x \wedge y$ denotes the largest common prefix of x and y.

The following two propositions will be useful to prove that observational and denotational semantics do coincide.

PROPOSITION 4.2. Unf(P) *and* $\mathscr{T}[\![P]\!]$ *are isomorphic.*

Proof. The proof proceeds by induction on the syntactic structure of P:

1. $P = 0$. The transition system associated to process 0, $LTS(0)$, is $\langle \{0\}, \varnothing, \varnothing \rangle$ and its unfolding is $\text{unf}(0) = \langle \varnothing, \varnothing, \varnothing \rangle$ that coincide with $\mathscr{T}[\![0]\!]$.

2. $P = 1$. The transition system associated to process 1, $LTS(1)$, is $\langle \{1\}, \{\langle 1, \varepsilon \rangle\}, \{1 \xrightarrow{\langle 1, \varepsilon \rangle} 1\} \rangle$ and its unfolding $\text{Unf}(1) = \langle \{x\}, \alpha(x) = \varepsilon, \beta(x, x) = \varepsilon \rangle$ with $x = 1 \xrightarrow{\langle 1, \varepsilon \rangle} 1$. The unfolding coincides with $\mathscr{T}[\![1]\!]$.

3. $P = P_1 + P_2$. Let $LTS(P_1) = \langle Z_1, L_1, T_1 \rangle$ and $LTS(P_2) = \langle Z_2, L_2, T_2 \rangle$ be the transition systems associated to P_1 and P_2, respectively. By induction hypothesis there exists an isomorphism f: $\text{Unf}(P_1) \to \mathscr{T}[\![P_1]\!]$ and an isomorphism

g: $\mathrm{Unf}(P_2) \to \mathscr{T}[\![P_2]\!]$ with $\mathrm{Unf}(P_1) = (X_1, \alpha_1, \beta_1)$ and $\mathrm{Unf}(P_2) = (X_2, \alpha_2, \beta_2)$. We have to show that there exists also an isomorphism h: $\mathrm{Unf}(P_1 + P_2) \to \mathscr{T}[\![P_2 + P_2]\!]$.

The transition system associated to $P_1 + P_2$ is $LTS(P_1 + P_2) = \langle Z, L, T \rangle$, where

$$Z = Z_1 - \{P_1\} \cup Z_2 - \{P_2\} \cup \{P_1 + P_2\}$$

$$L = L_1 \cup L_2$$

$$T = T_1 \cup \{P_1 + P_2 \xrightarrow{\mu, lu} P_1' \mid P_1 \xrightarrow{\langle \mu, u \rangle} P_1' \in T_1\} - \{P_1 \xrightarrow{\langle \mu, u \rangle} P_1'\}$$
$$\cup T_2 \cup \{P_1 + P_2 \xrightarrow{\langle \mu, ru \rangle} P_2' \mid P_2 \xrightarrow{\langle \mu, u \rangle} P_2' \in T_2\} - \{P_2 \xrightarrow{\langle \mu, u \rangle} P_2'\}.$$

Now, a transition sequence

$$P_1 + P_2 \xrightarrow{\langle \mu_1, lu_1 \rangle} P_1' \xrightarrow{\langle \mu_2, u_2 \rangle} \cdots \xrightarrow{\langle \mu_{n-1}, u_{n-1} \rangle} P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1$$
if a run of $LTS(P_1 + P_2)$ iff $P_1 \xrightarrow{\langle \mu_1, u_1 \rangle} P_1' \xrightarrow{\langle \mu_2, u_2 \rangle} \cdots \xrightarrow{\langle \mu_{n-1}, u_{n-1} \rangle} P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1$ if a run of $LTS(P_1)$.

Similarly, $P_1 + P_2 \xrightarrow{\langle \mu_1, ru_1 \rangle} P_2' \xrightarrow{\langle \mu_2, u_2 \rangle} \cdots \xrightarrow{\langle \mu_{n-1}, u_{n-1} \rangle} P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1$ is a run of $LTS(P_1 + P_2)$ iff $P_2 \xrightarrow{\langle \mu_1, u_1 \rangle} P_2' \xrightarrow{\langle \mu_2, u_2 \rangle} \cdots \xrightarrow{\langle \mu_{n-1}, u_{n-1} \rangle} P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1$ is a run of $LTS(P_2)$.

Thus, the unfolding of $P_1 + P_2$ is $\mathrm{Unf}(P_1 + P_2) = (X, \alpha, \beta)$, where

$$X = \{P_1 + P_2 \xrightarrow{\langle \mu_1, lu_1 \rangle} P_1' \cdots P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1 \mid P_1 \xrightarrow{\langle \mu_1, u_1 \rangle} P_1' \cdots P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1\}$$
$$\cup \{P_1 + P_2 \xrightarrow{\langle \mu_1, ru_1 \rangle} P_2' \cdots P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1 \mid P_2 \xrightarrow{\langle \mu_1, u_1 \rangle} P_1' \cdots P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1\}$$

$$\alpha(P_1 + P_2 \xrightarrow{\langle \mu_1, lu_1 \rangle} P_1' \cdots P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1) = \alpha_1(P_1 \xrightarrow{\langle \mu_1, u_1 \rangle} P_1' \cdots P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1);$$

$$\alpha(P_1 + P_2 \xrightarrow{\langle \mu_1, ru_1 \rangle} P_2' \cdots P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1) = \alpha_2(P_2 \xrightarrow{\langle \mu_1, u_1 \rangle} P_2' \cdots P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1);$$

$$\beta(P_1 + P_2 \xrightarrow{\langle \mu_1, lu_1 \rangle} P_1' \cdots P_{n_1-1} \xrightarrow{\langle \mu_{n_1}, u_{n_1} \rangle} 1, P_1 + P_2 \xrightarrow{\langle \mu_1', lu_1' \rangle} P_1'' \cdots P'_{n_2-1} \xrightarrow{\langle \mu_{n_2}, u_{n_2} \rangle} 1)$$
$$= \beta_1(P_1 \xrightarrow{\langle \mu_1, u_1 \rangle} P_1' \cdots P_{n_1-1} \xrightarrow{\langle \mu_{n_1}, u_{n_1} \rangle} 1, P_1 \xrightarrow{\langle \mu_1', u_1' \rangle} P_1'' \cdots P'_{n_2-1} \xrightarrow{\langle \mu_{n_2}, u_{n_2} \rangle} 1)$$

$$\beta(P_1 + P_2 \xrightarrow{\langle \mu_1, ru_1 \rangle} P_2' \cdots P_{n_1-1} \xrightarrow{\langle \mu_{n_1}, u_{n_1} \rangle} 1, P_1 + P_2 \xrightarrow{\langle \mu_1', ru_1' \rangle} P_2'' \cdots P'_{n_2-1} \xrightarrow{\langle \mu_{n_2}, u_{n_2} \rangle} 1)$$
$$= \beta_2(P_2 \xrightarrow{\langle \mu_1, u_1 \rangle} P_2' \cdots P_{n_1-1} \xrightarrow{\langle \mu_{n_1}, u_{n_1} \rangle} 1, P_2 \xrightarrow{\langle \mu_1', u_1' \rangle} P_2'' \cdots P'_{n_2-1} \xrightarrow{\langle \mu_{n_2}, u_{n_2} \rangle} 1)$$

$$\beta(P_1 + P_2 \xrightarrow{\langle \mu_1, lu_1 \rangle} P_1' \cdots P_{n_1-1} \xrightarrow{\langle \mu_{n_1}, u_{n_1} \rangle} 1, P_1 + P_2 \xrightarrow{\langle \mu_1', ru_1' \rangle} P_2'' \cdots P'_{n_2-1} \xrightarrow{\langle \mu_{n_2}, u_{n_2} \rangle} 1) = \varepsilon.$$

It is easy now to prove that function h: $\mathrm{Unf}(P_1 + P_2) \to \mathscr{T}[\![P_1]\!] \oplus \mathscr{T}[\![P_2]\!]$ defined by

$$h(P_1 + P_2 \xrightarrow{\langle \mu_1, lu_1 \rangle} P_1' \cdots P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1) = f(P_1 \xrightarrow{\langle \mu_1, u_1 \rangle} P_1' \cdots P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1),$$

$$h(P_1 + P_2 \xrightarrow{\langle \mu_1, ru_1 \rangle} P_2' \cdots P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1) = g(P_2 \xrightarrow{\langle \mu_1, u_1 \rangle} P_2' \cdots P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1)$$

is an isomorphism.

4. $P = P_1 \bullet P_2$. We assume active(P_1) and active(P_2). Otherwise the result is trivial. Thus let $LTS(P_1) = \langle Z_1, L_1, T_1 \rangle$ and $LTS(P_2) = \langle Z_2, L_2, T_2 \rangle$ be the transition systems associated to P_1 and P_2, respectively. By induction hypothesis there exists an isomorphism f: Unf(P_1) $\to \mathscr{T}[\![P_1]\!]$ and an isomorphism g: Unf(P_2) $\to \mathscr{T}[\![P_2]\!]$ with Unf(P_1) $= (X_1, \alpha_1, \beta_1)$ and Unf(P_2) $= (X_2, \alpha_2, \beta_2)$. We have to show that there exists also an isomorphism h: Unf($P_1 \bullet P_2$) $\to \mathscr{T}[\![P_2 \bullet P_2]\!]$. To do this we consider the transition system associated to $P_1 \bullet P_2$ and build h over the concatenations of runs of P_1 and runs of P_2 by exploiting isomorphism f and g. That is function h: Unf($P_1 \bullet P_2$) $\to \mathscr{T}[\![P_1]\!] \otimes \mathscr{T}[\![P_2]\!]$ defined by

$$h(P_1 \bullet P_2 \xrightarrow{\langle \mu_1, u_1 \rangle} P_1' \bullet P_2 \cdots P_{n-1}' \bullet P_2 \xrightarrow{\langle \mu_n, u_n \rangle} P_2'' \cdots P_m'' \xrightarrow{\langle \mu_n, u_m \rangle} 1)$$
$$= \langle f(P_1 \xrightarrow{\langle \mu_1, u_1 \rangle} P_1' \cdots P_{n-1} \xrightarrow{\langle 1, u_n^1 \rangle} 1), g(P_2 \xrightarrow{\langle \mu_n, u_n^2 \rangle} P_2'' \cdots P_m'' \xrightarrow{\langle \mu_m, u_m \rangle} 1) \rangle,$$

where $P_1 \xrightarrow{\langle \mu_1, u_1 \rangle} P_1' \cdots P_{n-1}' \xrightarrow{\langle 1, u_n^1 \rangle} 1$ is a run of P_1 and $P_2 \xrightarrow{\langle \mu_n, u_n^2 \rangle} P_2'' \cdots P_m'' \xrightarrow{\langle \mu_m, u_m \rangle} 1$ is a run of P_2 is the wanted isomorphism. ▮

PROPOSITION 4.3. *$P \lhd_r Q$ if and only if there exists a strict monomorphism f: Unf(P) $\to$ Unf(Q).*

Proof. ($\Rightarrow$). The proof proceeds by induction on weight(P). Assume $P \lhd_r Q$. We show that there exists a strict monomorphism f: Unf(P) $\to$ Unf(Q). First, we consider the case 1-actions are performed by P, i.e. if $P \xrightarrow{1} M$ implies $Q \xrightarrow{1} M'$ and $\exists i$ injective: $M \to M'$, s.t. $\forall P' \in M$, $P' \lhd_r f(P')$. Now $P \xrightarrow{1} M$ with $|M| = n_1$ iff $P \xrightarrow{\langle 1, u_1 \rangle} 1, \ldots, P \xrightarrow{\langle 1, u_{n_1} \rangle} 1$ and $Q \xrightarrow{1} M'$ with $|M'| = n_2$ iff $Q \xrightarrow{\langle 1, u_1' \rangle} 1, \ldots, Q \xrightarrow{\langle 1, u_{n_1}' \rangle} 1, \ldots, Q \xrightarrow{\langle 1, u_{n_2}' \rangle}$. We let: $f(P \xrightarrow{1, u_1} 1) = Q \xrightarrow{1, u_1'} 1, \ldots, f(P \xrightarrow{1, un_1} 1) = Q \xrightarrow{1, u_{n_1}'} 1$.

We consider now the case of a generic a-transition from P; i.e., $P \xrightarrow{a} M$ implies $Q \xrightarrow{a} M'$ and $\exists i$ injective: $M \to M'$, s.t. $\forall P' \in M$, $P' \lhd_r f(P')$.

W.l.o.g. we can assume $|M| = n_1$ and $|M'| = n_2$ with $n_1 \leqslant n_2$ and consider the n_1 transitions performed by P and matched by injection i. That is, $P \xrightarrow{\langle a, u_1 \rangle} P_1'$ is matched by $Q \xrightarrow{\langle a, u_1' \rangle} Q_1'$ and $P_1' \lhd_r Q_1' = i(P_1')$, ..., $P \xrightarrow{\langle a, u_{n_1} \rangle} P_{n_1}'$ is matched by $Q \xrightarrow{\langle a, u_{n_1}' \rangle} Q_{n_1}'$ and $P_{n_1}' \lhd_r Q_{n_1}' = i(P_{n_1}')$.

By induction hypothesis there are strict monomorphims h_1: Unf(P_1') $\to$ Unf(Q_1'), ..., h_{n_1}: Unf(P_{n_1}') $\to$ Unf(Q_{n_1}'). Now, for every run from P, we let

$$f(P \xrightarrow{\langle a, u_1 \rangle} P_1' \xrightarrow{\langle \mu_2, u_1 \rangle} P_2 \cdots P_{n-1} \xrightarrow{\langle \mu_n, u_n \rangle} 1) = Q \xrightarrow{\langle a, u_1' \rangle} h_1(P_1' \xrightarrow{\langle \mu_2, u_2 \rangle} P_2 \cdots \xrightarrow{\langle \mu_n, u_n \rangle} 1)$$
$$\vdots \vdots \vdots$$
$$f(P \xrightarrow{\langle a, u_{n_1} \rangle} P_{n_1}' \xrightarrow{\langle \mu_2, u_2 \rangle} P_2 \cdots P_{n-1} \xrightarrow{\langle \mu_n\, u_n \rangle} 1) = Q \xrightarrow{\langle a, u_{n_1}' \rangle} h_{n_1}(P_{n_1}' \xrightarrow{\langle \mu_2, u_2 \rangle} P_2 \cdots \xrightarrow{\langle \mu_n, u_n \rangle} 1).$$

It is easy to show that f is a strict monomorphism.

($\Leftarrow$). The proof proceeds again by induction on weight(P). Assume there exists a strict monomorphism f: Unf(P) $\to$ Unf(Q). We prove that is there exists an *r-simulation* $\mathfrak{R}$ containing the pair (P, Q).

First, we consider the case P performs a 1-transition $P \xrightarrow{1} M$ with $|M| = n_1$. Thus, there are n_1 transitions $P \xrightarrow{\langle 1, u_1 \rangle} 1, \ldots, P \xrightarrow{\langle 1, u_{n_1} \rangle} 1$. Let us examine the possible

1-transitions from Q, $Q \xrightarrow{1} M'$, with $|M'| = n_2$. Thus, there are n_2 transitions $Q \xrightarrow{\langle 1, u'_1 \rangle} 1, \ldots, Q \xrightarrow{\langle 1, u'_{n_2} \rangle}$. Clearly these transitions are runs from Q, and since f: $\mathrm{Unf}(P) \to \mathrm{Unf}(Q)$ is a strict monomorphism and, hence, injective, it follows that $n_1 \leqslant n_2$. Thus, there exists $\exists i$ injective: $M \to M'$. The wanted relation is $\mathfrak{R}_0 = \{(1, 1)\}$ if $n_1 \neq 0$.

Consider now the case in which a generic a-transition is performed by P, $P \xrightarrow{a} M$. We prove that there exists a transition from Q, $Q \xrightarrow{a} M'$ and $\exists i$ injective: $M \to M'$, s.t. $\forall P' \in M$, $P' \lhd_r f(P')$.

Let us see the actual transitions performed by P: $\xrightarrow{\langle a, u_1 \rangle} P'_1, \ldots, P \xrightarrow{\langle a, u_{n1} \rangle} P'_{n_1}$ and $M = \{\!|P'_1, \ldots, P'_{n_1}|\!\}$. Consider, for instance, all runs of $\mathrm{Unf}(P)$, starting with transition $P \xrightarrow{\langle a, u_1 \rangle} P'_1$:

$$
\begin{array}{lll}
x_1 = P \xrightarrow{\langle a, u_1 \rangle} P'_1 \xrightarrow{\langle \mu_1^1, u_{11} \rangle} P_2^1, \ldots, P_{n_1 - 1}^1 \xrightarrow{\langle \mu_1^{n_1}, u_{1n_1} \rangle} P_{n_1}^1 \\
\vdots \quad \vdots \qquad\qquad\qquad\qquad \vdots \\
x_k = P \xrightarrow{\langle a, u_1 \rangle} P'_1 \xrightarrow{\langle \mu_k^1, u_{k1} \rangle} P_2^k, \ldots, P_{n_k - 1}^k \xrightarrow{\langle \mu_k^{n_k}, u_{1n_k} \rangle} P_{n_k}^k
\end{array}
$$

and note that the agreement between these runs is at least the initial a.

Since f: $\mathrm{Unf}(P) \to \mathrm{Unf}(Q)$ is a strict monomorphism there are $y_1, \ldots, y_k \in \mathrm{Unf}(Q)$ such that $f(x_1) = y_1, \ldots, f(x_k) = y_k$. Moreover, a strict monomorphism preserves both extension and agreement; thus every run y_i starts with a transition $Q \xrightarrow{\langle a, u'_1 \rangle} Q'_1$ because the agreement between $y_1, \ldots, y_k$ has at least the initial a. Clearly, the strict correspondence between $x_1, \ldots, x_k$ and $y_1, \ldots, y_k$ implies that there exists a strict monomorphism between $\mathrm{Unf}(P'_1)$ and $\mathrm{Unf}(Q'_1)$ and, thus, by induction hypothesis that $P'_1 \lhd_r Q'_1$. Let $\mathfrak{R}_{(a, u_1)}$ be the *r-simulation* containing (P'_1, Q'_1).

Clearly, we can repeat the above reasoning for all the other transitions $P \xrightarrow{\langle a, u_2 \rangle} P'_2, \ldots, P \xrightarrow{\langle a, u_{n_1} \rangle} P'_{n_1}$, obtaining relations $\mathfrak{R}_{(a, u_2)}, \ldots, \mathfrak{R}_{(a, u_{n1})}$. We denote $\mathfrak{R}_a = \mathfrak{R}_{(a, u_1)} \cup \mathfrak{R}_{(a, u_2)} \cup \cdots \cup \mathfrak{R}_{(a, u_{n1})}$. Again, we can repeat the reasoning for all μ-transitions that P can perform obtaining relations $\mathfrak{R}_{\mu_1}, \ldots, \mathfrak{R}_{\mu_k}$. It is easy now to prove that relation $\mathfrak{R} = \mathfrak{R}_a \cup \mathfrak{R}_{\mu_1} \cup \cdots \cup \mathfrak{R}_{\mu_k}$ is a *r-simulation* for (P, Q). ∎

By Proposition 4.2 and 4.3 it follows that

THEOREM 4.4. *$P \lhd_r Q$ if and only if there exists a strict monomorphism f: $\mathscr{T}[\![P]\!] \to \mathscr{T}[\![Q]\!]$.*

5. MODELS OF INFINITE EXPRESSIONS

We consider now the language with the star operator. To avoid considering terms leading to infinitely branching trees, we restrict attention to terms without iterations of 1 within *-contexts (that is, terms that do not have the empty word property [30]). This is necessary for proving some crucial properties of our relations, e.g. that the existence of two inverse monomorphisms implies isomorphism. The wanted property is defined by a boundedness predicate.

TABLE 9

(_)*-Axioms for $\leqslant$

	$1 + X \bullet X^* \leqslant X^*$	(*1)
Let	$X_0 = 1$ $X_{n+1} = 1 + X \bullet X_n$	
$\forall n \in N$	$X_n \bullet Z \leqslant Y$ implies $X^* \bullet Z \leqslant Y$	(Ω-R)

DEFINITION 5.1 (Nonempty word property and boundedness predicates). Let *nwp* and *bound* be the least predicates over **PL** terms that satisfy

- nwp(a), nwp(0);
  - — nwp(P) and nwp(Q) implies nwp($P + Q$);
  - — nwp(P) or nwp(Q) imply nwp($P \bullet Q$)
- bound(a), bound(0), and bound(1);
  - — bound(P) and bound(Q) imply bound($P + Q$) and bound($P \bullet Q$)
  - — bound(P) and nwp(P) implies bound(P^*).

The nesting degree of a **PL** process P is defined as the maximum number of nested (_)* contexts.

DEFINITION 5.2 (Nesting degree). The nesting degree-of a P, nd(P), is defined by the inference rules:

- $\mathrm{nd}(0) = \mathrm{nd}(1) = \mathrm{nd}(a) = 0$
- $\mathrm{nd}(P + Q) = \mathrm{nd}(P \bullet Q) = \max\{\mathrm{nd}(P), \mathrm{nd}(Q)\}$
- $\mathrm{nd}(P^*) = \mathrm{nd}(P) + 1$.

Like for the operators considered in the previous section, we can give a denotational, an observational, and an axiomatic account of $(-)^*$. In order to prove that the three views do coincide, we rely on the two rules of Table 9 that we will prove sound for both the denotational and the observational semantics. One of the rules is an ω-induction rule, the other (*1) is borrowed from Table 2. We have not been able to prove completeness by relying only on the two rules of the latter, and had to resort to (ω − R). This rule can be used to derive the other rule (*2), of Table 2.

5.1. A Denotational Account of Kleene Star

We start by defining the objects of our denotational model.

DEFINITION 5.3. Given a tree $t = (X, \alpha, \beta)$ over the alphabet A (see Definition 2.2), we can define $t^\infty = (X^\infty, \alpha^\infty, \beta^\infty)$:

1. $X^\infty = \{\langle x_1, x_2, ..., x_n\rangle \mid n \in N \text{ and } x_i \in X\}$
2. $\alpha^\infty(\langle x_1, x_2, ..., x_n\rangle) = \alpha(x_1)\,\alpha(x_2)\,\cdots\alpha(x_n)$

3. $\beta^{\infty}(\langle x_1, x_2, ..., x_n\rangle, \langle y_1, y_2, ..., y_m\rangle)$

$$= \begin{cases} \alpha(x_1)\,\alpha(x_2)\cdots\alpha(x_k)\,\beta(x_{k+1}, y_{k+1}) \\ \qquad \text{if} \quad x_i = y_i, \forall i, 0 \leqslant i \leqslant k; x_{i+1} \neq y_{i+1}; \\ \alpha(x_1)\,\alpha(x_2)\cdots\alpha(x_k \\ \qquad \text{if} \quad x_i = y_i, \forall i, 0 \leqslant i \leqslant k; n = k \text{ or } m = k. \end{cases}$$

Obviously trees of the form t^{∞} are no longer finite even if t is finite and their runs are chains of runs; category $\mathbf{T}^{\text{fin}}$ can be extended to encompass these infinite objects and the corresponding morphisms, by introducing colimits for chains of the form: $t^0 \to t^1 \to t^2 \to t^3 \to \cdots \to t^{j-1} \to t^j \to t^{j+1} \to \cdots$, where t^j is defined below and morphisms are the strict monomorphisms given by the obvious inclusions.

DEFINITION 5.4. The semantic approximant $t^j = (X^j, \alpha^j, \beta^j)$ of t is defined by

— $X^j = \{\langle x_1, x_2, ..., x_n\rangle \mid x_i \in X, 0 \leqslant n \leqslant j\}$

— $\alpha^j(\langle x_1, x_2, ..., x_n\rangle) = \alpha(x_1)\,\alpha(x_2)\cdots\alpha(x_n)$

— $\beta^j(\langle x_1, x_2, ..., x_n\rangle, \langle y_1, y_2, ..., y_m\rangle)$

$$= \begin{cases} \alpha(x_1)\,\alpha(x_2)\cdots\alpha(x_k)\,\beta(x_{k+1}, y_{k+1}) \\ \qquad \text{if} \quad x_i = y_i, \forall i, 0 \leqslant i \leqslant k; x_{i+1} \neq y_{i+1}; \\ \alpha(x_1)\,\alpha(x_2)\cdots\alpha(x_k \\ \qquad \text{if} \quad x_i = y_i, \forall i, 0 \leqslant i \leqslant k; n = k \text{ or } m = k. \end{cases}$$

LEMMA 5.5. *If $T = \mathscr{T}[\![P]\!]$ for some P in* **PL** *then $t_j = \mathscr{T}[\![P_j]\!]$, where P_j is the* syntactic approximant *defined as in Table* 9.

Proof. It requires an easy induction on j. ∎

The new category $\mathbf{T}^{\infty}$ is generated by closing $\mathbf{T}^{\text{inf}}$ with respect to $(-)^{\infty}$ and to the operators introduced in Section 2.

DEFINITION 5.6. A tree $t = (X, \alpha, \beta) \in \mathbf{T}^{\infty}$ is image finite if for any $w \in A^{\star}$, the set $X_w = \{x \in X \mid \alpha(x) = w\}$, is finite.

We let $Tree^{\text{if}}$ be the class of image finite trees within $\mathbf{T}^{\infty}$. For the class of systems we consider requiring image finiteness amounts to requiring trees to be finitely branching. Indeed, within our framework a tree $t = (X, \alpha, \beta) \in \mathbf{T}^{\infty}$ is *finitely branching* if, once we let Y be a subset of X such that for every x, $y \in Y$, $\beta(x, y) = w$, $w \in A^{\star}$, we have that Y is finite. It is not difficult to see, by structural induction, that over **PL** image finiteness implies finite branching.

The algebraic interpretation via function $\mathscr{T}$ can now be extended to deal with the star operator.

DEFINITION 5.7. An algebraic interpretation for $(-)^*$ in $Tree^{\text{if}}$ via function $\mathscr{T}$ is given by $\mathscr{T}[\![P^*]\!] = \mathscr{T}[\![P]\!]^{\infty}$.

PROPOSITION 5.8. *$\mathscr{T}[\![P]\!] \in Tree^{\text{if}}$ for all $P \in$* **PL**.

Proof. The proof goes by structural induction. Boundedness is crucial. The only nontrivial case is dealing with P^*. We have that $\mathscr{T}[\![P^*]\!] = (X^{\infty}, \alpha^{\infty}, \beta^{\infty})$ whenever

$\mathscr{T}[\![P]\!] = (X, \alpha, \beta)$. Thus, we have that the length of $\alpha(x)$ ($|\alpha(x)|$) is bigger than or equal to 1, $\forall x \in X$. Because of this, we have that $y \in X^\infty$ and $|\alpha^\infty(y)| = n$ imply that $y \in X^n$. The claim now follows from image finiteness of all approximants. ▮

We now prove that if, for each $n \in N$, there exists a strict monomorphism between an approximant t_n of t^∞ and a tree s, then there exists a strict monomorphism between t^∞ and s. This property will be crucial to prove soundness of ω-induction rule.

LEMMA 5.9. *Let t and s be trees in $Tree^{\text{if}}$. If for each $j \in N$ there exists a strict monomorphism $f_j: t_j \to s$, then there exists a strict monomorphism $f^\infty: t^\infty \to s$.*

Proof. Every tree in $Tree^{\text{if}}$ has countable many runs. Hence, there exists an enumeration of them. Let us take the first run, x_1, and look for its image. We have a nonempty finite set of possibilities because it is contained in at least one approximation. Let us consider the second run, x_2, and look for an image of x_1 and x_2 strictly preserving their agreement. This is again possible because there exists at least an approximation which contains both runs with the same agreement. In this procedure we could be forced to change the image of x_1 in order to satisfy the conditions on agreement. Note that, in this case, the former image of x_1 is not usable anymore in the future. Let us now suppose to found the image till x_n. Try to find the image for x_{n+1}. Following similar reasonings of above, there surely exists a possible choice for x_{n+1}.

This procedure stops for every run, because whenever we have to change its image, we definitely discard a run in S. In fact we have finitely many runs of the required extent. Hence the image of a run is fixed after finitely many steps of procedure.

By construction the resulting function is indeed a strict monomorphism. ▮

PROPOSITION 5.10. *If $t = (X, \alpha, \beta)$ and $t' = (X', \alpha', \beta')$ are two image finite trees such that there exist two strict monomorphisms $m: t \to t'$ and $m': t' \to t$ then t and t' are isomorphic.*

Proof. Due to our image finiteness assumption, we have that, for all $w \in A^*$, the set $X_w = \{x \in X \mid \alpha(x) = w\}$, is finite. It is injectively mapped via m into X'_w that is again finite. The same reasoning can be applied if we start with m'. This suffices to establish existence of a bijection between X and X' that, since m and m' are morphisms, extends to an isomorphism between t and t'. ▮

The above result shows that strict monomorphisms can be used to define a partial order over $Tree^{\text{if}}$ and thus enables us to state the following proposition.

PROPOSITION 5.11. *t^∞ is the least fixed point for the endofunctor $F_t: Tree^{\text{if}} \to Tree^{\text{if}}$ defined as $F_t([\,-\,]) = t \otimes [\,-\,] + \mathbf{1}$.*

Proof. F_t satisfies the co-completeness condition required in order to have a minimal fixed point [32], i.e. $F_t(\text{colim}\langle F_t^n(\mathbf{0}), F_t^n(\mathbf{0}_{F_t(\mathbf{0})})\rangle) \equiv \text{colim}\, F_t(\langle F_t^n(\mathbf{0}), F_t^n(\mathbf{0}_{F_t(\mathbf{0})})\rangle)$. In fact, $t^\infty \equiv F_t(t^\infty)$ is such an object and, therefore, the least fixed point. Obviously, $\equiv$ stands for "is isomorphic to." ▮

Axiom (*1) and rule (*2) of Table 2 are a direct consequence of the above result. We are left with establishing soundness of ω-induction rule.

PROPOSITION 5.12. *ω-induction rule is sound for the denotational interpretation.*

Proof. Follows directly from Lemma 5.9 and Lemma 5.5. ▮

DEFINITION 5.13 (Head normal forms). A head normal form is either 0 or a term of the form

$$\left(\sum_{i\in I} a_i + \sum_{j\in J} a_j \bullet P_j\right) + \sum_{k\in K} Q_k,$$

where for all k, $Q_k = 1$ and every P_j is a process different from 0 and 1.

LEMMA 5.14. *Every finitely branching term, P, can be transformed into a head normal form,* $\mathrm{hnf}(P)$, *by using the laws of Table* 6 *and axiom* $X^* = 1 + X \bullet X^*$.

Proof. The proof follows similar lines of that of Lemma 2.10. The only additional case to consider is $P = S^*$. By axiom $X^* = 1 + X \bullet X^*$ we have $S^* = 1 + S \bullet S^*$ and, hence, $\mathrm{hnf}((S^*)) = 1 + \mathrm{hnf}(S \bullet S^*)$. Note that $\mathrm{bound}(S^*)$ implies that $\mathrm{hnf}(S)$ is of the form $\sum_{i\in I} a_i + \sum_{j\in J} a_j \bullet S_j$ and, thus, that R^* terms cannot appear at the top level in $\mathrm{hnf}(S \bullet S^*)$. ▮

Another kind of normal forms will be also useful in the sequel. They are normal forms in which processes P^* are considered as atoms.

DEFINITION 5.15 (Finite normal forms). A finite normal form is either 0 or a term of the form

$$\left(\sum_{i\in I} a_i + \sum_{j\in J} n_j^* + \sum_{k\in K} a_k \bullet n_k\right) + \sum_{l\in L} n_l^* \bullet n_l' + \sum_{m\in M} P_m,$$

where $P_m = 1$ for all m and n_j, n_k, n_l, n_l' are finite normal forms different from 0 and 1. Moreover, $\mathrm{bound}(n_j^*)$ and $\mathrm{bound}(n_l^*)$.

LEMMA 5.16 (Reduction to finite normal forms). *Every finite* **PL** *term P is provably equal, via the laws of Table* 6, *to a finite normal form* $\mathrm{fnf}(P)$.

Proof. Similar to Lemma 2.10. ▮

Completeness of our proof system with respect to the tree-based model, relies on a preliminary lemma. Intuitively, it states that if we can syntactically prove that $P_2 \leqslant Q'$, by assuming existence of a strict monomorphism $\mu: \mathscr{T}[\![P_2]\!] \to \mathscr{T}[\![Q']\!]$ for some process Q', then we can also deal with terms of the form $R_2 \bullet P_2$.

LEMMA 5.17. *Let P_2 be a* **PL** *process such that for every* **PL** *process Q' and strict monomorphism $\mu': \mathscr{T}[\![\mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q')]\!]$ we have* $\mathrm{fnf}(P_2) \leqslant \mathrm{hnf}(Q')$. *Let R and Q be* **PL** *processes with the latter enjoying the property that there exists a strict monomorphism $\mu: \mathscr{T}[\![R \bullet \mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q)]\!]$. Then $R \bullet \mathrm{fnf}(P_2) \leqslant \mathrm{hnf}(Q)$.*

Proof. The proof proceeds by induction on $\mathrm{nd}(R)$. First, we consider the base case: $\mathrm{nd}(R)=0$. Here, we now proceed by induction on the syntactic structure of term R:

- $R=0$. Then $R \bullet \mathrm{fnf}(P_2) \leqslant \mathrm{hnf}(Q)$ because we have the inequation $0 \leqslant Q$ (by axiom (S5) we also have $0 \bullet \mathrm{fnf}(P_2)=0$).
- $R=1$. By axiom (S3) there is a strict monomorphism $\mu'\colon \mathscr{T}[\![\mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q)]\!]$ and, hence, by hypothesis $\mathrm{fnf}(P_2) \leqslant \mathrm{hnf}(Q)$. By axiom (S3) the thesis follows.
- $R=a$. Then $\mu\colon \mathscr{T}[\![a \bullet \mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q)]\!]$. Thus $\mathrm{hnf}(Q)$ contains a summand of the form $a \bullet S$ such that there exists a strict monomorphism $\mu'\colon \mathscr{T}[\![\mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(S)]\!]$. By hypothesis $\mathrm{fnf}(P_2) \leqslant \mathrm{hnf}(S)$ and, hence, $a \bullet \mathrm{fnf}(P_2) \leqslant a \bullet \mathrm{hnf}(S) \leqslant \mathrm{hnf}(Q)$.
- $R=n_1+n_2$ (with $\mathrm{nd}(n_1)=0$ and $\mathrm{nd}(n_2)=0$). Then $\mu\colon \mathscr{T}[\![(n_1+n_2) \bullet \mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q)]\!]$. Thus also $\mu\colon \mathscr{T}[\![n_1 \bullet \mathrm{fnf}(P_2)+n_2 \bullet \mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q)]\!]$ and, hence, there are two strict monomorphisms μ_1 and μ_2 such that $\mu_1\colon \mathscr{T}[\![n_1 \bullet \mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q_1)]\!]$ and $\mu_2\colon \mathscr{T}[\![n_2 \bullet \mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q_2)]\!]$ with $\mathrm{hnf}(Q)=\mathrm{hnf}(Q_1)+\mathrm{hnf}(Q_2)$. By structural induction $n_1 \bullet \mathrm{fnf}(P_2) \leqslant \mathrm{hnf}(Q_1)$ and $n_2 \bullet \mathrm{fnf}(P_2) \leqslant \mathrm{hnf}(Q_2)$. Thus also $(n_1+n_2) \bullet \mathrm{fnf}(P_2) \leqslant \mathrm{hnf}(Q_1)+\mathrm{hnf}(Q_2)=\mathrm{hnf}(Q)$.
- $R=a \bullet n_j$ (with $\mathrm{nd}(n_j)=0$). In this case then $\mu\colon \mathscr{T}[\![a \bullet n_j \bullet \mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q)]\!]$. By structural induction follows that whenever $\mu\colon \mathscr{T}[\![n_j \bullet \mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q')]\!]$ for some **PL** process Q', then $n_j \bullet \mathrm{fnf}(P_2) \leqslant \mathrm{hnf}(Q')$. Let $\mathrm{fnf}(n_j \bullet \mathrm{fnf}(P_2))$ be the finite normal form of $n_j \bullet \mathrm{fnf}(P_2)$. Always by structural induction whenever $\mu\colon \mathscr{T}[\![a \bullet \mathrm{fnf}(n_j \bullet \mathrm{fnf}(P_2))]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q)]\!]$ is $a \bullet \mathrm{fnf}(n_j \bullet \mathrm{fnf}(P_2)) \leqslant \mathrm{hnf}(Q)$.
- For the other syntactic options ($R=S^*$ and $R=S^* \bullet n_j$), we cannot have $\mathrm{nd}(R)=0$.

Now, we proceed with the inductive step and assume the claim true for $\mathrm{nd}(R)<n$. Again by induction on the syntactic structure of term R we prove that it holds also for $\mathrm{nd}(R)=n$. Cases $R=0$, $R=1$, $R=a$ are not possible because $\mathrm{nd}(R)=0$. Cases $R=n_1+n_2$ and $R=a \bullet n_j$ follow by similar lines as above. The only critical cases are $R=S^*$ and $R=S^* \bullet n_j$:

- $R=S^*$. Then $\mu\colon \mathscr{T}[\![S^* \bullet \mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q)]\!]$. This and the definition of $\mathscr{T}[\![S^*]\!]$ imply existence of strict monos $\mathscr{T}[\![S_i \bullet \mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q)]\!]$ for every i. Since $\mathrm{nd}(S_i)<\mathrm{nd}(S^*)$, by induction on the nesting degree, follows $S_i \bullet \mathrm{fnf}(P_2) \leqslant \mathrm{hnf}(Q)$. By ω-induction rule follows $S^* \bullet \mathrm{fnf}(P_2) \leqslant \mathrm{hnf}(Q)$.
- $R=S^* \bullet n_j$ (with $\mathrm{nd}(S^*) \leqslant n$ and $\mathrm{nd}(n_k) \leqslant n$). In this case then $\mu\colon \mathscr{T}[\![S^* \bullet n_j \bullet \mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q)]\!]$. By structural induction follows that whenever $\mu\colon \mathscr{T}[\![n_j \bullet \mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q')]\!]$ for some **PL** process Q', then $n_j \bullet \mathrm{fnf}(P_2) \leqslant \mathrm{hnf}(Q')$. Let $\mathrm{fnf}(n_j \bullet \mathrm{fnf}(P_2))$ be the finite normal form of $n_j \bullet \mathrm{fnf}(P_2)$. Always by structural induction whenever $\mu\colon \mathscr{T}[\![S^* \bullet \mathrm{fnf}(n_j \bullet \mathrm{fnf}(P_2))]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q)]\!]$ is $S^* \bullet \mathrm{fnf}(n_j \bullet \mathrm{fnf}(P_2)) \leqslant \mathrm{hnf}(Q)$. ∎

THEOREM 5.18. *Let P, Q be* **PL** *processes and t, s be trees such that $t=\mathscr{T}[\![P]\!]$ and $s=\mathscr{T}[\![Q]\!]$. If there exists a strict monomorphism $\mu\colon t \to s$, then $P \leqslant Q$.*

Proof. The proof proceeds by induction on the syntactic structure of P:

- $\mathrm{fnf}(P) = 0$. $\mathscr{T}[\![0]\!] \to \mathscr{T}[\![Q]\!]$ for any Q, but we also have the inequation $0 \leqslant Q$.
- $\mathrm{fnf}(P) = 1$. If $\mathscr{T}[\![1]\!] \to \mathscr{T}[\![Q]\!]$ then $\mathscr{T}[\![Q]\!]$ contains a run with extent equal to ε, but this means that 1 is a summand of Q.
- $\mathrm{fnf}(P) = a$: $\mathscr{T}[\![a]\!] \to \mathscr{T}[\![Q]\!]$ implies that $\mathrm{hnf}(Q)$ contains a summand of the form $a \bullet (1 + R)$. It is easy to prove that $a \leqslant a \bullet (1 + R)$.
- $\mathrm{fnf}(P) = \mathrm{fnf}(P_1) + \mathrm{fnf}(P_2)$. Assume now that $\mathscr{T}[\![\mathrm{fnf}(P_1) + \mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q)]\!]$; since we have restricted ourselves to finitely branching trees and can rely on associativity and commutativity of $+$, we have that $\mathrm{hnf}(Q) = \mathrm{hnf}(Q_1) + \mathrm{hnf}(Q_2)$ with $\mathscr{T}[\![\mathrm{hnf}(P_i)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q_i)]\!]$; $i = 1, 2$. The claim follows from the inductive hypothesis.
- $\mathrm{fnf}(P) = \mathrm{fnf}(P_1) \bullet \mathrm{fnf}(P_2)$. Assume now that $\mathscr{T}[\![\mathrm{fnf}(P_1) \bullet \mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q)]\!]$; we have that $\mathrm{fnf}(P_1)$ is either a generator, say a, or a term of the form R^*. Since by structural induction $\mathscr{T}[\![\mathrm{fnf}(P_2)]\!] \to \mathscr{T}[\![\mathrm{hnf}(Q')]\!]$ implies $\mathrm{fnf}(P_2) \leqslant \mathrm{hnf}(Q')$ for every Q', the thesis follows by Lemma 5.17.
- $\mathrm{fnf}(P) = R^*$. Apply Lemma 5.17 to process $R^* \bullet 1$ and then axiom (S2) of Table 1. ∎

The main theorem is a direct consequence of the above considerations.

THEOREM 5.19. *Let $\mathscr{T}[\![\mathbf{PL}]\!]$ be the set of trees in Tree$^\infty$ obtained by interpreting elements of* **PL**. *($\mathscr{T}[\![\mathbf{PL}]\!]$, $\oplus$, $\otimes$, **0**, **1**, $(-)^\infty$) ordered via strict monomorphisms is the free model for the set of axioms* (C1)–(C3), (S1)–(S5), *and* (RD) *of Table* 1, *the laws of Table* 4, *axiom* (*1) *and ω-induction of Table* 9.

5.2. An Observational Account of Kleene Star

The operational semantics for *bound* infinite terms is described in Table 10. We extend the predicate *active* of Table 7 to *-terms by asserting: active(P^*).

We now establish soundness of the ω-induction rule with respect to r-simulation. In order to do this, we need some definitions and preliminary results. First of all, we define the "satisfiability predicate." It is denoted by S and relates pairs of processes (P, Q) to pairs of actions and natural numbers (μ, n). $S(P, Q, \mu, n)$ holds if and only if, starting with μ-labelled transitions from P, process Q can (r-)simulate P for at most n-steps of simulation.

Formally predicate $S(P, Q, \mu, n)$ with $P, Q \in \mathbf{PL}$, $\mu \in A \cup \{1\}$, and $n \in N$ is defined inductively by:

- $S(P, Q, \mu, 1)$ iff $P \xrightarrow{\mu} M$ implies $Q \xrightarrow{\mu} M'$ and $\nexists f$ injective: $M \to M'$;
- $S(P, Q, \mu, \max\{n_1, ..., n_k\} + 1)$ iff $P \xrightarrow{\mu} M$ implies $Q \xrightarrow{\mu} M'$ and each f_i injective: $M \to M'$, $i \in [1 \cdots k]$, there exists $P_i \in M$, $\mu_i \in A \cup \{1\}$ such that $S(P_i, f_i(P_i), \mu, n_i)$.

PROPOSITION 5.20. *Let P and Q be* **PL** *processes. If $P \lhd_r Q$ then here exists an action $\mu \in A \cup \{1\}$ and $n \in N$ such that $S(P, Q, \mu, n)$.*

TABLE 10

Operational Semantics of (_)*

$$(\mathrm{Star}_1)\ \frac{}{P^* \xrightarrow{\langle 1, \varepsilon\rangle} 1}, \quad (\mathrm{Star}_2)\ \frac{P \xrightarrow{\langle \mu, u\rangle} P'}{P^* \xrightarrow{\langle \mu, u\rangle} P' \bullet P^*}$$

Proof. We show that

$$\mathfrak{R} = \{(P_1, Q_1) \mid \forall \mu \in A \cup \{1\}, \forall n \in N, \neg S(P_1, Q_1, \mu, n)\}$$

is a resource simulation. To prove that $\mathfrak{R}$ is a r-simulation we have to prove that for each $(P_1, Q_1) \in \mathfrak{R}$, $P_1 \xrightarrow{\mu} M$ implies $Q_1 \xrightarrow{\mu} M'$ and $\exists f$ injective: $M \to M'$, s.t. $\forall P' \in M$, $\langle P', f(P')\rangle \in \mathfrak{R}$. By contradiction suppose that there exists a pair $(P_1, Q_1) \in \mathfrak{R}$ such that $P_1 \xrightarrow{\mu} M$ and $Q_1 \xrightarrow{\mu} M'$ but $\forall g$ injective: $M \to M'$, there exists $R \in M$ for which $\langle R, g(R)\rangle \notin \mathfrak{R}$. This implies that for each injection $g_1, ..., g_k : M \to M'$ there exists a process $R_i \in M$, an action $\mu_i \in A \cup \{1\}$ and a step $n_i \in N$ such that $S(R_i, g_i(R_i), \mu_i, n_i)$. But then $S(P_1, Q_1, \mu, \max\{n_1, ..., n_k\} + 1)$, and, hence $(P_1, Q_1) \notin \mathfrak{R}$ by contradicting the hypothesis. ∎

We can now introduce two lemmas that permit understanding the relationships between infinite behaviours and their approximants. They will be instrumental to prove soundness of the ω-induction rule.

LEMMA 5.21. *Let n, k be natural numbers such that $k \leqslant n$ and let $[1 + P \bullet (1 + P \bullet (\cdots (1 + P \bullet P^*) \cdots))] \bullet R$ be obtained by $P^* \bullet R$ by applying $P^* \to 1 + P \bullet P^*$ n times. Then for every computation from $P^* \bullet R \xrightarrow{\langle \mu_1, u_1\rangle} P_1 \cdots P_{k-1} \xrightarrow{\langle \mu_k, u_k\rangle} P_k$ there exists a computation from $[1 + P \bullet (1 + P \bullet (\cdots (1 + P \bullet P^*) \cdots))] \bullet R \xrightarrow{\langle \mu_1, u_1\rangle} P'_1 \cdots P'_{k-1} \xrightarrow{\langle \mu_k, u_k\rangle} P'_k$, and vice versa.*

Proof. The fact that $P^* \xrightarrow{\langle \mu, u\rangle} P'$ iff $1 + P \bullet P^* \xrightarrow{\langle \mu, u\rangle} P'$ and an inspection of the operational rules are sufficient to prove the claim. ∎

LEMMA 5.22. *Let n, k be natural numbers such that $k \leqslant n$ and $P_n \bullet R = [1 + P \bullet (1 + P \bullet (\cdots (1 + P) \cdots))] \bullet R$. Then for every computation from $P_n \bullet R \xrightarrow{\langle \mu_1, u_1\rangle} P''_1 \cdots P''_{k-1} \xrightarrow{\langle \mu_k, u_k\rangle} P''_k$ with $k \leqslant n$, there exists a computation from $P^* \bullet R \xrightarrow{\langle \mu_1, u_1\rangle} P_1 \cdots P_{k-1} \xrightarrow{\langle \mu_k, u_k\rangle} P_k$ and vice versa.*

Proof. Similar to that for Lemma 5.21. ∎

Notation. Processes P''_i and P_i in Lemma 5.22 will be called *correspondent* processes.

PROPOSITION 5.23. *The ω-induction rule is sound for the observational interpretation.*

Proof. Suppose, by contradiction, that $\forall n \in N\ P_n \bullet R \lhd_r Q$ but $P^* \bullet R \not\lhd_r Q$. By Proposition 5.20 then there exists $\mu \in A \cup \{1\}$ and $n \in N$ such that $S(P^* \bullet R, Q, \mu, n)$.

Consider now transition $P_n \bullet R \xrightarrow{\mu} M_1^d$ By Lemma 5.22 $P^* \bullet R \xrightarrow{\mu} M_1$ and $|M_1^s| = |M_1|$ because every process $P''_1 \in M_1^d$, obtained by $P_n \bullet R \xrightarrow{\langle \mu_1, u_1\rangle} P''_1$, has a

(unique) correspondent $P_1 \in M_1$ such that $P^* \bullet R \xrightarrow{\langle \mu_1, u_1 \rangle} P_1$ and vice versa. By hypothesis $P_n \bullet R \lhd_r Q$ thus $Q \xrightarrow{\mu} M'_1$ and there exists an injection $g_1 : M_1^d \to M'_1$ such that for each $d_1^i \in M_1^d$, $d_1^i \lhd_r g_1(d_1^i)$. Clearly g_1 detects an injection $f_1 : M_1 \to M'_1$ such that $g_1(d') = f_1(P')$, where $P' \in M_1$ is the correspondent process of $d' \in M_1^d$.

From $S(P^*R, Q, \mu, n)$, it follows that there are $P_1 \in M_1$, $\mu_1 \in A \cup \{1\}$, $n_1 \in N$ with $n_1 \leqslant n$ such that $S(P_1, f_1(P_1), \mu_1, n_1)$. Consider now P_1 and the correspondent d_1 in M_1^d. We have that $g_1(d_1) = f_1(P_1)$ and $d_1 \lhd_r g_1(d_1)$. Hence, $d_1 \xrightarrow{\mu^1} M_2^d$ and $P_1 \xrightarrow{\mu^1} M_2$ such that $|M_2^d| = |M_2|$, because every process $d_2^i \in M_2^d$ is a correspondent of some process in M_2 relative to the same actual $\langle \mu_1, u \rangle$-transition and vice versa. From $d_1 \lhd_r g_1(d_1)$ follows that $g_1(d_1) = f_1(P_1) \xrightarrow{\mu_1} M'_2$ and there exists an injection $g_2 : M_2^d \to M'_2$ such that for each $d_2^i \in M_2^d$, $d_2^i \lhd_r g_2(d_2^i)$. Now g_2 permits us to detect an injection $f_2 : M_2 \to M'_2$, such that $g_2(d') = f_2(P')$ and $P' \in M_2$ is the correspondent of d' in M_2^d. Moreover since $S(P_1, f_1(P_1), \mu_1, n_1)$ there are $P_2 \in M_2$, $\mu_2 \in A \cup \{1\}$, $n_2 \in N$ with $n_2 \leqslant n_1$ such that $S(P_2, f_2(P_2), \mu_2, n_2)$.

By following the above reasoning, we obtain $d_n \lhd_r g_n(d_n)$ and $S(P_n, f_n(P_n), \mu_n, 1)$.

Suppose now that $d_n \xrightarrow{\mu_n} M_{n+1}^d$ and $P_n \xrightarrow{\mu_n} M_{n+1}$ with $|M_{n+1}^d| = |M_{n+1}|$ where every $d_{n+1}^i \in M_{n+1}^d$ is the correspondent of some process $P_{n+1} \in M_{n+1}$ relative to the same actual $\langle \mu_{n+1}, u \rangle$-transition. By hypothesis we know that $d_n \lhd_r g_n(d_n)$ and that injection g_n permits to detect an injection $f_n : M_n \to M'_n$, such that $g_n(d') = f_n(P')$ with d' is the correspondent in M_n^d of $P' \in M_n$. Thus, we have that $g_n(d_n) = f_n(P_n) \xrightarrow{\mu_n} M'_{n+1}$ and that there exists an injection $g_{n+1} : M_{n+1}^d \to M'_{n+1}$ such that for each $d_{n+1}^i \in M_{n+1}^d$, $d_{n+1}^i \lhd_r g_{n+1}(d_{n+1}^i)$. This contradicts the fact that $S(P_n, f_n(P_2), \mu_n, 1)$. ▌

PROPOSITION 5.24 (Completeness). *Axioms* (C1)–(C3), (S1)–(S5), *and* (RD) *of Table* 1, *the laws of Table* 4, *axiom* (*1) *and* ω*-induction of Table* 9, *soundly and completely axiomatize r-simulation over full* **PL**.

Proof. Soundness of the axioms in Table 1 have been established in Section 3 and of (*1) can be proved by exhibiting a simulation containing the pair $\langle 1 + P \bullet P^*, P^* \rangle$; the proof of this exploits 5.21. Soundness of the ω-induction rule has been established in Proposition 5.23.

The completeness proof is completely similar to that given for the denotational model (Theorem 5.18); obviously, in the new proof, we have to let r-simulations play the rôle that strict monomorphisms were playing in the old one. ▌

The proof that observational and denotational semantics are sound and complete with respect to the same set of axioms and ω-induction rule, guarantees that the two models coincide. Actually, the proof that for each P and $Q \in$ **PL** we have

$$\mathscr{T}[\![P]\!] \leqslant \mathscr{T}[\![Q]\!] \quad \text{iff} \quad P \lhd_r Q$$

can be given in a more direct way by observing at ω-induction allows us to reason about approximants of P with a smaller nesting degree. Again, a proof similar to that of Theorem 5.18 is needed.

6. FURTHER WORK

There are different directions along which one could think of extending the present work. Here, apart for mentioning the importance of looking for finitary axiomatizations of our interpretation of regular expressions (see, e.g. [2, 31]), we would like to discuss the line of research that is dictated by those developed in the context of process algebras. We shall consider:

- the impact of extending **PL** with operators for parallel composition;
- the introduction of "silent" actions that leads to so-called weak equivalences;
- the search for a temporal logic that "agrees" with resource bisimulation.

6.1. Dealing with Parallelism

We can further extend our language with the binary operator $|$ that can be interpreted as parallel composition. The parallel operator that naturally pops up is one that permits pairs of concurrent processes to progress only if both can perform the same actions; see, e.g. [22]. We will name **PPL**, for parallel **PL**, the language obtained by extending **PL** with this new operator.

Also for **PPL** we can define an operational and a denotational semantics and show that they coincide. The two new semantics are obtained from the old ones by adding a clause for the new operator; see Table 11 and the denotational interpretation of $P \mid Q$ at the end of these subsection. Also the original axiomatization is extended to **PPL** by adding a small set of laws for the new operator.

For the operational semantics, we also need to slightly modify the definition of the *active* predicate in order to decide whether active$(P \mid Q)$ holds. This is due to the fact that, to model parallel composition of two processes P and Q, we need to ensure that the two processes can perform a common maximal trace. This property can be checked by determining the sequences of actions they can perform (Lang(P) and Lang(Q)) and by requiring that their intersection be nonempty. This guarantees that $\exists s = \mu_1 \mu_2 \cdots \mu_n$ such that $P \xrightarrow{\langle \mu_1, u_1 \rangle} \cdots \xrightarrow{\langle \mu_n, u_n \rangle} 1$ and $Q \xrightarrow{\langle \mu_1, u'_1 \rangle} \cdots \xrightarrow{\langle \mu_n, u'_n \rangle} 1$. From this we can infer that also $P \mid Q$ successfully terminates; i.e., $P \mid Q \xrightarrow{\langle \mu_1, u_1 \bullet u'_1 \rangle} \cdots \xrightarrow{\langle \mu_n, u_n \bullet u'_n \rangle} 1$. Formally, Lang$(P)$ is defined by adding the rule Lang$(P \mid Q) =$ Lang$(P) \cap$ Lang(Q) to the set of rules that permit associating a language to a regular expression.

TABLE 11

Operational Semantics for PPL

$$(\mathrm{Par}_1) \quad \frac{P \xrightarrow{\langle a, u \rangle} P',\ Q \xrightarrow{\langle a, u' \rangle} Q',\ \mathrm{active}(P' \mid Q')}{P \mid Q \xrightarrow{\langle a, u \mid u' \rangle} P' \mid Q'},$$

$$(\mathrm{Par}_2) \quad \frac{P \xrightarrow{\langle 1, u \rangle} 1,\ Q \xrightarrow{\langle 1, u' \rangle} 1}{P \mid Q \xrightarrow{\langle 1, u \mid u' \rangle} 1}$$

TABLE 12

A Complete Set of Axioms for Parallelism

$X \mid Y = Y \mid X$		(Par1)
$(X \mid Y) \mid Z = X \mid (Y \mid Z)$		(Par2)
$X \mid (Y + 2) = X \mid Y + X \mid Z$		(Par3)
$X \mid 0 = 0$		(Par4)
$\mu \bullet X \mid \mu' \bullet Y = \mu \bullet (X \mid Y)$	if $\mu = \mu'$	(Par5)
$\mu \bullet X \mid \mu' \bullet Y = 0$	if $\mu \neq \mu'$	(Par6)
$1 \mid 1 = 1$		(Par7)

It is possible to show that for each **PPL** term, P, there exists a finite state automata that accepts Lang(P). Moreover, the set of our languages is closed with respect to complementation and union and it is decidable whether Lang(P) $\cap$ Lang(Q) is empty or not. This permits us to define active($P \mid Q$) by Lang(P) $\cap$ Lang(Q) $\neq \varnothing \Rightarrow$ active($P \mid Q$). Also for the richer language, we have a complete axiomatization of r-simulation; the new axioms are reported in Table 12.

PROPOSITION 6.1 (Completeness). *The axioms for* **PL** *and those in Table* 12 *completely axiomatize r-simulation over* **PPL**.

As promised, we can provide also a denotational interpretation of the parallel combinator. We take advantage of the following property of **T**.

PROPOSITION 6.2. **T** *has products.*

Proof. Given $t_1 = (X_1, \alpha_1, \beta_1)$ and $t_2 = (X_2, \alpha_2, \beta_2)$, $t_1 \times t_2 = \langle X, \alpha, \beta \rangle$ is defined by

- $X = \{\langle x_1, x_2 \rangle \in X_1 \times X_2 \mid \alpha_1(x_1) = \alpha_2(x_2)\}$
- $\alpha(\langle x_1, x_2 \rangle) = \alpha_1(x_1) = \alpha_2(x_2)$
- $\beta(\langle x_1, x_2 \rangle, \langle y_1, y_2 \rangle) = \min(\beta_1(x_1, y_1), \beta_2(x_2, y_2))$. ∎

We can now extend our algebraic interpretation of **PL** to **PPL** by defining

$$\mathscr{T}[\![P \mid Q]\!] = \mathscr{T}[\![P]\!] \times \mathscr{T}[\![Q]\!]$$

and again obtain a model for the full system of axioms.

6.2. Weak Equivalences

In this section we sketch how our work can be extended to languages with invisible (τ-)actions; i.e., we study the "weak" versions of resource simulation and resource bisimulation. The language **PL** is extended with the basic process τ that performs the silent action τ and successfully terminates:

$$\tau \xrightarrow{\langle \tau, \varepsilon \rangle} 1.$$

To define silent transitions, we will rely on a new relation $\mapsto^{\tau}$ that permits observing the branching structure of terms by "signaling" those states that are *real choice*

points, i.e. those states that have at least two *active* (see Table 7) alternatives. Relation $\overset{\tau}{\mapsto}\tau$ is used to describe the execution of consecutive τ actions (possibly interleaved by 1's actions) with no real choice required. Clearly, we should make sure that every process of the form $P+Q$ with $\neg\text{active}(Q)$ has $\overset{\tau}{\mapsto}$-transitions whenever P has $\overset{\tau}{\mapsto}$-transitions. This is because we want to equate, for instance, processes $\tau+0$ and τ. Using the *active* predicate in the definition of $\overset{\tau}{\mapsto}$ enables us to detect such situations.

A weak silent transition $\overset{\langle\tau,\rangle}{\Longrightarrow}$ is either $\overset{\tau}{\mapsto}$ transition (and, hence, $u=\varepsilon$), or a sequence of $\xrightarrow{\langle\tau,u_i\rangle}$, with choice sequence u_i different from ε, preceded and followed by $\overset{\tau}{\mapsto}$ transitions ($\overset{\tau}{\mapsto}\xrightarrow{\langle\tau,u_i\rangle}\overset{\tau}{\mapsto}$). In the latter case the choice sequence is obtained by concatenating those of the involved single step transitions.

As usual a visible weak transition $\overset{\langle a,u\rangle}{\Longrightarrow}$ is a transition $\xrightarrow{\langle a,u'\rangle}$ possibly preceded and followed by invisible weak transitions. Thus, $\overset{\langle a,u\rangle}{\Longrightarrow}=\overset{\langle\tau,u_1\rangle}{\Longrightarrow}\xrightarrow{\langle a,u_2\rangle}\overset{\langle\tau,u_3\rangle}{\Longrightarrow}$ and $u=u_1u_2u_3$.

To fix intuition, let us concentrate on two processes:

- $P=\tau\bullet\tau\bullet(\tau\bullet\tau\bullet b+\tau\bullet\tau\bullet\tau\bullet c)$,
- $(\tau\bullet\tau\bullet b+\tau\bullet\tau\bullet c)$.

Process P has a single $\overset{\tau}{\mapsto}$-transition: $P\overset{\tau}{\mapsto}(\tau\bullet\tau\bullet b+\tau\bullet\tau\bullet c)$ while process Q has no $\overset{\tau}{\mapsto}$-transition.

Now, if we concentrate on P, we have that it has three $\overset{\langle\tau,u\rangle}{\Longrightarrow}$-transitions:

1. $P\overset{\langle\tau,\varepsilon\rangle}{\Longrightarrow}(\tau\bullet\tau\bullet b+\tau\bullet\tau\bullet c)$,
2. $P\overset{\langle\tau,l\rangle}{\Longrightarrow}b$,
3. $P\overset{\langle\tau,r\rangle}{\Longrightarrow}c$,

and two possible $\overset{\langle a,u\rangle}{\Longrightarrow}$-transitions:

1. $P\overset{\langle b,l\rangle}{\Longrightarrow}1$,
2. $P\overset{\langle c,r\rangle}{\Longrightarrow}1$.

Weak resource simulation and bisimulation can be obtained by replacing $\xrightarrow{\mu}$ with $\overset{\mu}{\Rightarrow}$ in Definition 3.2; the second arrow is obtained from $\overset{\langle\mu,u\rangle}{\Longrightarrow}$ just like the first one is obtained from $\xrightarrow{\langle\mu,u\rangle}$.

The resulting relations will equate $\tau\bullet\tau$ with τ and differentiate $\tau+\tau$ from τ. The reason for the latter differentiation is similar to that behind $1+1\not\sim_r 1$. Indeed $(\tau+\tau)\bullet a$ is equal to $\tau\bullet a+\tau\bullet a$ (by axiom (RD)) which has to be different (in this new setting) by $\tau\bullet a$.

Also weak resource simulation and bisimulation are preserved by all operators of **PL**; noticeably, they are preserved by $+$. This is another interesting property of our relations. Indeed, weak equivalences are usually not preserved by $+$ and additional work is needed to isolate the coarsest congruence contained in them. Here we take advantage of the new $\overset{\tau}{\mapsto}$-transition, to model initial τ-moves differently than the others.

We have completely axiomatized weak resource bisimulation over **PL**. The set of axioms is the same as that for resource bisimulation (see Table 1) with the addition of the simple τ-law in Table 13 with $\alpha\in A\cup\{\tau\}$. We would like to conclude by

TABLE 13

Axiom for Weak Resource Bisimulation

$\alpha \bullet \tau \bullet X = \alpha \bullet X$

remarking that Proposition 3.8 holds also in this new setting; i.e., weak resource bisimulation is the kernel of weak resource simulation. We are still working on the complete axiomatization of weak resource simulation. This and the impact of parallel and hiding operators on the weak setting will be the subject of further investigation.

6.3. Graded Modalities

A well-known result relating operational and logical semantics is that reported in [21]. In that paper, a modal logic, now known as Hennessy–Milner Logic (HML), is, defined which, when interpreted over labelled transition systems with (without) silent actions, is proved to be in full agreement with weak (strong) observational equivalence. Other correspondences have been established in [7], where two equivalences over Kripke structures (node-labelled transition systems) are related to two variants of CTL* [17], and in [16], where another variant of bisimulation called *branching bisimulation* is provided with three logical characterizations.

A logical characterization can be provided also for resource bisimulation. The new logics can be obtained by replacing both the box and diamond modality of HML with a family of graded modalities [18], defined below, where $\#$ denotes *multisets cardinality*,

$$p \models \langle \mu \rangle_n \psi \quad \text{if and only if} \quad \#\{\!|p' \mid p \xrightarrow{\mu} p';\ p' \models \psi|\!\} = n.$$

Now, if we define graded HML (GHML) as the set of formulae generated by the grammar

$$\psi ::= \text{True} \mid \text{False} \mid \psi_1 \vee \psi_2 \mid \psi_1 \wedge \psi_2 \mid \langle \mu \rangle_n \psi,$$

where μ is in A and $0 \leqslant n < \infty$, it can be established that

$$\forall \psi \in \text{GHML},\ P \models \psi \Leftrightarrow Q \models \psi \quad \text{if and only if} \quad P \sim_r Q.$$

7. CONCLUSION AND RELATED WORKS

Regular expressions have been the subject of many investigations since the early fifties. By now, they have three standard equivalent semantics that represent alternative ways of capturing the fact that regular expressions are a natural and compact notation for describing sets of (alternative) sequential behaviors of systems. Regular expressions have been equipped with:

- an *algebraic semantics* given in terms of a small set of simple and intuitive laws (Kleene algebras);
- a *denotational semantics* that associates sets of traces over a given alphabet of action to each regular expression;
- an *operational semantics* described in terms of equivalence classes (based on the set of accepted traces) of finite state automata.

In the paper, we have pursued the same program with a different interpretation of regular expressions in mind. We aim at using them also for describing interactive systems. Within this class of systems, nondeterminism plays a central rôle. We have studied alternative semantics that stress more the presence of choices and have shown that, also, in this case we can propose three alternative views of regular expressions:

- An *algebraic semantics* that is obtained from the original one by dropping two axioms, namely idempotence of $+$ and distribution of $\bullet$ over $+$ (nondeterministic Kleene algebras);
- A *denotational semantics* based on a natural class of labelled trees that are the free model for the nondeterministic axiomatization;
- an *operational semantics* described in terms of equivalence classes (based on a simulation preorder) of labelled transition systems.

The simulation (*r-simulation*) we use is new and takes resources into account; i.e. it counts the number of choices that processes still have after they have decided the specific action they intend to perform.

This new relation has two important features. First, we have that double r-simulation coincides with r-bisimulation. This enables us to study basic properties of the model by concentrating on the preorder and to use this behavioral relation also for stepwise refinements of systems implementation.

Second, in a setting with unobservable τ-actions, the *weak* version of the equivalence is a congruence (is preserved also by $+$) and can be axiomatized by simply adding to nondeterministic Kleene algebras the axiom: $\alpha \bullet \tau \bullet X = \alpha \bullet X$.

Kleene star-like operators and their axiomatization have been studied also in many other papers; however, we have to say that most of them take as a starting point the classical notions of strong and weak bisimulation and either do not consider the full language of regular expressions or do not have a free denotational tree-model for them. Fokkink and Zantema [19, 20] provide finite equational axiomatizations of standard strong bisimulation equivalence over process algebras with the binary variant of the star operator originally introduced by Kleene (23). P^*Q is used to denote the iteration of P followed by Q. This alternative iteration operator turns out to be indispensable for axiomatizing bisimulation when the calculus does not contain a neutral element for sequentialization. The work of Fokkink and Zantema has been extended in [1, 3, 4] to weak bisimulation.

Benson and Tiuryn [11] take as a starting point the axioms for bisimulation, rather than those of regular expressions and have results similar to ours, but they have to consider trees with two types of nodes, open and closed, and they introduce a single neutral element that plays the rôle of both 1 and 0.

Along the same line of research we can place the paper by Bloom, Ésik, and Taubner [8]. There, the relation of Milner's synchronization trees with Elgot's iteration theories is studied to show that synchronization trees have a finite axiomatization over iteration theories, but the model as it stands is not "initial" with respect to the considered axioms but needs to be factored via some basic axioms.

ACKNOWLEDGMENTS

The paper has benefitted from discussions with Michele Boreale, from suggestions by Luca Aceto and Davide Sangiorgi, and from a thorough reading and detailed comments by Zoltan Ésik. We thank them all, but keep for ourselves the blame for all remaining inaccuracies.

REFERENCES

1. L. Aceto, W. Fokkink, R. Glabbeek, and A. Ingolfsdottir, Axiomatizing prefix iteration with silent steps, *Inform. and Comput.* **197**, No. 1 (1996), 26–40.
2. L. Aceto, W. Fokkink, and A. Ingolfsdottir, "A Managerie of Non-finitely Based Process Semantics over BPA*: From Ready Simulation to Completed Trances," Research Report, BRICS, RS-96-23 1996. [To appear in *Mathematical Structures in Computer Science*]
3. L. Aceto and A. Ingolfsdottir, "A Complete Equational Axiomatization for Prefix Iteration with Silent Steps," Research Report RS-95-5, BRICS, 1995.
4. L. Aceto and A. Ingolfsdottir, An equational axiomatization of observation congruence for prefix iteration, *in* "Proceedings AMAST '96," LNCS, Vol. 1101, pp. 195–209, Springer-Verlag, New York/Berlin, 1996.
5. J. C. M. Baeten and J. A. Bergstra, Process algebra with a zero object, *in* "Concur '90," LNCS, Vol. 458, pp. 83–98, 1990.
6. L. Bernatsky, S. L. Bloom, Z. Esik, and Gh. Stefanescu, Equational theories of relations and regular sets, *in* "Proc. Conf. Words, Languages and Combinatorics, Kyoto, 1992," pp. 40–48, World Scientific, Singapore, 1994.
7. M. C. Browne, E. Clarke, and O. Grümberg, Characterizing Finite Kripke Structures in Propositional Temporal Logic, *Theoretical Computer Science* **59**, Nos. 1, 2 (1988), 115–131.
8. S. L. Bloom, Z. Ésik, and D. Taubner, Iteration theories of synchronization trees, *Information and Computation* **102** (1993), 1–55.
9. J. A. Bergstra and J. W. Klop, Process theory based on bisimulation semantics, *in* "LNCS," Vol. 354, pp. 50–122, 1989.
10. M. Boffa, Une remarque sur les systems complets d'identites rationelles, *Theoret. Inform. Appl.* **24** (1990), 419–423.
11. D. B. Benson and J. Tiuryn, Fixed points in free process algebras, Part 1, *Theoretical Computer Science* **63** (1989), 274–294.
12. J. Baeten and P. Weijland, "Process Algebras," Cambridge University Press, 1990.
13. F. Corradini, R. De Nicola, and A. Labella, Fully abstract models for nondeterministic regular expressions, *in* "Proc. Concur 95," LNCS, Vol. 962, pp. 130–144, Springer-Verlag, New York/Berlin, 1995.
14. R. De Nicola and A. Labella, Tree Morphisms and Bisimulations, *in* "Proc. MFCS '98 Workshop on Concurrency," Electronics Notes in Theoretical Computer Science, Vol. 18, 1998.
15. R. De Nicola and A. Labella, A completeness theorem for nondeterministic kleene algebras, *in* "MFCS '94," LNCS, Vol. 841, pp. 536–545, 1994.
16. R. De Nicola and F. Vaandrager, Three logics for branching bisimulation, *J. Assoc. Comput. Mach.* **33** (1986), 151–178.

17. E. H. Emerson and Y. Halpern, "Sometimes" and "not never" revisited: On branching versus linear time temporal logic, *Journal of ACM* **42** (1995), 458–487.
18. M. Fattorosi-Barnaba and F. De Caro, Graded Modalities, I; *Studia Logica* **44** (1985), 197–221.
19. W. Fokkink, A complete equational axiomatization for prefix iteration, *Inform. Process. Lett.* **52** (1994), 333–337.
20. W. Fokkink and H. Zantema, Basic process algebra with iteration: completeness of its equational axioms, *Comput. J.* **37** (1994), 259–267.
21. M. Hennessy and R. Milner, Algebraic Laws for Nondeterminism and Concurrency, *J. Assoc. Comput. Mach.* **32** (1985), 137–161.
22. C. A. R. Hoare, "Communicating Sequential Processes," Prentice–Hall, Englewood Cliffs, NJ, 1989.
23. S. C. Kleene, Representation of Events in Nerve Nets and Finite Automata, *in* "Automata Studies" (Shannon and McCarthy, Eds.), pp. 3–41, Princeton Univ. Press, 1956.
24. S. Kasangian and A. Labella, Enriched Categorical Semantics for Distributed Calculi, *J. Pure Appl. Algebra* **83** (1992), 295–321.
25. D. Kozen, A completeness theorem for kleene algebras and the algebra of regular events, *Information and Computation* **110** (1994), 366–390.
26. D. Krob, Complete systems of B-rational identities, *Theoretical Computer Science* **89** (1991), 207–343.
27. R. Milner, "Communication and Concurrency," Prentice–Hall, 1989.
28. D. Park, Concurrency and automata on infinite sequences, *in* "Proc. GI," LNCS, Vol. 104, pp. 167–183, 1981.
29. B. C. Pierce, "Basic Category Theory for Computer Scientists," The MIT Press, Cambridge, MA, 1991.
30. A. Salomaa, Two complete axiom systems for the algebra of regular events, *Journal of ACM* **13** (1966), 158–169.
31. P. Sewell, Bisimulation is not finitely (first order) equationally axiomatisable, *in* "Proc. of LICS," IEEE Press, 1994.
32. M. B. Smith and G. D. Plotkin, The category-theoretic solution of recursive domain equation, *J. Comput.* **11** (1982), 762–783.